WO0064189

Publication Title:

SAFE COLOR LIMITING OF A COLOR ON A DIGITAL NONLINEAR EDITING SYSTEM

Abstract:

Abstract of WO0064189

A system and method of transforming a component color into safe color space. A color component of a first color space is received. The component color is transformed into safe color space by applying conversion terms to each component of the component color. The transformation into safe color space creates symmetrical relationships between the color components. These symmetrical relationships permit simplified operations to be performed on the component. The transformation may be used to generate safe color from components of a pixel according to a threshold defining safe color, where components include a luma component, a first chroma component, and a second chroma component. It is determined whether a combination of a first luma value associated with the luma component and a first chroma value associated with the first and second chroma components of the pixel exceeds the threshold defining safe color. If the combination exceeds the threshold defining safe color, one or more components of the pixel is modified to generate safe components.

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : H04N 9/64

A2

(11) International Publication Number: **WO 00/64189**

(43) International Publication Date: 26 October 2000 (26.10.00)

(21) International Application Number: PCT/US00/10082

(22) International Filing Date: 14 April 2000 (14.04.00)

(30) Priority Data:
09/293,590 16 April 1999 (16.04.99) US

(71) Applicant: AVID TECHNOLOGY, INC. [US/US]; Avid Technology Park, One Park West, Tewksbury, MA 01876 (US).

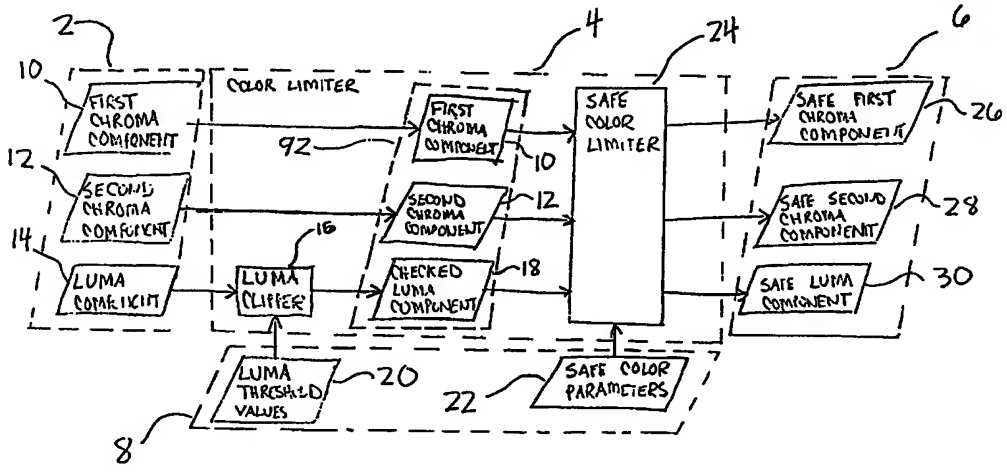
(72) Inventors: CACCIATORE, Raymond, D.; 5 Nonset Lane, Westford, MA 01886 (US). LAIRD, Michael, D.; 38 Westgate Crossing, Nashua, NH 03062 (US).

(74) Agent: GORDON, Peter, J.; Wolf, Greenfield & Sacks, P.C., 600 Atlantic Avenue, Boston, MA 02210 (US).

(81) Designated States: AU, CA, JP, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: SAFE COLOR LIMITING OF A COLOR ON A DIGITAL NONLINEAR EDITING SYSTEM



(57) Abstract

A system and method of transforming a component color into safe color space. A color component of a first color space is received. The component color is transformed into safe color space by applying conversion terms to each component of the component color. The transformation into safe color space creates symmetrical relationships between the color components. These symmetrical relationships permit simplified operations to be performed on the component. The transformation may be used to generate safe color from components of a pixel according to a threshold defining safe color, where components include a luma component, a first chroma component, and a second chroma component. It is determined whether a combination of a first luma value associated with the luma component and a first chroma value associated with the first and second chroma components of the pixel exceeds the threshold defining safe color. If the combination exceeds the threshold defining safe color, one or more components of the pixel is modified to generate safe components.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

SAFE COLOR LIMITING OF A COLOR ON A DIGITAL NONLINEAR EDITING SYSTEM

BACKGROUND

Digital non-linear editing (DNLE) is a process by which digital media may be edited. DNLE, as the name implies, is performed on digital media stored as data in digital media files on a digital random access medium. DNLE may be conducted in a non-linear fashion because the digital media files in which the digital media is stored can be randomly accessed. Thus an editor may access a piece of the digital media without having to proceed sequentially through other pieces of the digital media stored in the same or other digital media files. More than one editor also may be able to access different pieces of the same digital media contemporaneously. The digital media may be a digitized version of a film or videotape or digital media produced through live capture onto a disk of a graphics or animation software application. Example commercial DNLE systems include the Media Composer® or the Avid® Symphony™ video production system or NewsCutter® news editing system available from Avid Technology, Inc. For a more detailed description of DNLE, see *Digital Nonlinear Editing, New Approaches to Editing Film and Video*, 1993 , by Thomas Ohanian.

Digital video is a form of digital media typically processed on a DNLE. Digital video is comprised of a sequence of digital images. Digital images are comprised of an array of picture cells called pixels. DNLE systems often process digital video at multiple rates. For example, the video may correspond to a playback rate of 30 (more precisely 29.97) frames per second (fps) in accordance with the NTSC (National Television Standards Committee) standards, or 25 fps in accordance with PAL (phase alternate line) standards. The digital video may be captured from film which typically has a rate of 24 frames per second. The United States of America and Japan have generally adopted the NTSC standards and European nations have generally adopted the PAL standards. There are multiple other formats for high definition and digital television signals.

One frame defines a complete video image that can be displayed on a television screen. Frames include a series of odd and even lines of pixels. Frames are divided into two fields of information, where a first field includes all the odd lines of pixels, and a second field includes all the even lines of pixels. For example, for an NTSC frame typically including 525 lines, a first field includes pixels of lines 1, 3, 5, . . ., 525, and the second field includes pixels of lines 2, 4, 6, . . ., 524. When the analog video signal is

broadcast to a television, the fields are alternatively scanned and drawn to the television screen. This alternate scanning is called interlacing. Formats using images that are not interlaced are called progressive.

A variety of data formats can be used to represent the color of pixels within a digital image. Formats may be classified into two major categories: composite signals and component signals. Component formats represent a color as multiple components, each component defining a value along a dimension of the color space in which the color being represented is defined. A composite video is an analog signal that uses a high frequency subcarrier to encode color information with a signal that encodes luminance information. The subcarrier is a sinewave of which the amplitude is modulated by the saturation of the color represented by the signal, and the hue of the color is encoded as a phase difference from a color burst. Analog composite signals are generally used to broadcast television video signals.

There are a variety of component formats used to represent color. RGB (Red, Green, Blue) format represents a color with a red component, a green component and a blue component. CMYK (Cyan, Magenta, Yellow, Black) format represents a color with a cyan component, a magenta component, and a yellow component. CMYK is a format commonly used by printers. The CMYK components are color opposites of RGB components. In a three-dimensional coordinate system, each component of either the RGB or the CMYK format represents a value along an axis, the combination of the values defining a cubic color space.

The data formats HSL (Hue, Saturation, Lightness or Luminance) and HSV (Hue, Saturation, Value) represent a color with a hue component, a saturation component, and a luminance component. In a three-dimensional coordinate system, the luma component represents a value along a luma axis, the hue component represents the angle of a chroma vector with respect to the luminance axis and the saturation component represents the magnitude of the chroma vector. The combination of the values defines a hexagonal cone-shaped color space.

YCrCb, YUV, and YIQ are three formats that represent a color with a luma component Y, and two chroma components, Cr and Cb, U and V, or I and Q, respectively, that define a chroma vector. In a three-dimensional coordinate system, each component of either the YCrCb, YUV, and YIQ format represents a value along an axis, the combination of the values defining a cylindrical color space around the luma

axis. The chroma components define the chroma vector. In data formats with a luminance component, the luma component can be used independently to represent a pixel in a black and white image to be displayed, for example, with a black and white monitor.

YCrCb is a format used for digital video signals; YUV and YIQ are formats used for analog video signals. A standard YcrCb format is the 4:2:2 Y_{601}CrCb format (CCIR 601 format), where the luma component, Y_{601}, of the pixels of the component signal are transmitted at twice the frequency of the chroma components, Cr and Cb (hence the notation 4:2:2). Only every other pixel (even pixel) is represented with chroma components. For example, the components of the pixels, P0, P1, P2, P3, etc., of a 601 format signal are transmitted as follows: Cr0,Y0,Cb0, Y1, Cr2, Y2, Cb2, Y3, etc. When the video image is displayed, the chroma components of the odd pixels are determined by averaging, combining, interpolating, etc., the chroma components of the preceding even pixel and the next even pixel. The 4:2:2 sampling rate is used to save bandwidth on the broadcast medium, and also saves storage space when storing the signal.

The analog component YUV format is the format from which an analog composite signal may be constructed. The relationship between the YUV components and the values of the NTSC analog composite sine wave are as follows: Amplitude = $[0.0925 \cdot (Y+C) + 0.075]$,

where $C = V \cdot \sin(\omega t + 0.576) + U \cdot \cos(\omega t + 0.576)$, and $\omega = 3.579545$ MHz.

When components of a component signal are combined into a composite signal, it is possible for some component combinations to push the signal amplitude above or below some threshold value. When the frequency band of the signal is too wide, the color is considered "unsafe", because it may cause interference with a television signal having a different carrier frequency. The component signal is corrected such that the resulting composite signal is within a "safe region." Various methods have been developed to correct a component signal such that a composite signal constructed from the component signal is within a safe region.

SUMMARY

One problem with current techniques for safe color correction of a pixel is that the resulting corrected color may not be the closest color in the three-dimensional color space to the original unsafe color. This problem may occur if the three components of

the signal are not reduced proportionately. Typically less than all three components are corrected or clipped. In particular, the luminance component often is not changed.

When a corrected pixel is displayed along with proximate pixels that were not corrected, or other pixels which were corrected, artifacts may be seen in a portion of an image. These artifacts may be reduced by modifying the luminance component. By modifying an unsafe color such that both the luma component and the chroma components are modified, the unsafe color is mapped to the closest safe color in the color space and such artifacts are further reduced.

Also, unsymmetrical relationships between the components of a component color may involve complex operations to be perfomed on the components to determine if the color is safe, and to determine the modifications to be applied to an unsafe color to make it a safe color. Safe color modification provides a method of transforming a component color into a safe color space. A component color of a first color space is received. The component color is transformed into a safe color space by applying conversion terms to each component of the component color. The transformation into a safe color space creates symmetrical relationships between the color components, which permits simplified operations to be performed on the components.

In one aspect, a method is provided of generating components for a pixel to provide safe color from components of the pixel according to a threshold defining safe color. The components include a luma component, a first chroma component, and a second chroma component. It is determined whether a combination of a first luma value associated with the luma component and a first chroma value associated with the first and second chroma components of the pixel exceeds the threshold defining safe color. If the combination exceeds the threshold defining safe color, the luma component of the pixel is modified to generate safe components.

In an embodiment, if the combination exceeds the threshold defining safe color, the first and second chroma components of the pixel are modified to generate components within the threshold defining safe color.

In another aspect, provided is a method of generating correction values for modifying components of a pixel to provide a safe color according to a threshold defining safe color. The color components are transformed into safe space. It is determined whether a combination of the transformed color components exceeds the threshold defining safe color. If the combination exceeds the threshold defining safe

color, component correction values are generated to be applied to the components and produce safe components.

BRIEF DESCRIPTION OF THE DRAWINGS

In the drawings,

Fig. 1 is a data flow diagram illustrating an embodiment of a system for safe color modification;

Fig. 2 is a data flow diagram illustrating an embodiment of the system of Fig. 1 in more detail;

Fig. 3 is a data flow diagram illustrating an embodiment of the operation of a luma threshold converter;

Fig. 4 is a flow chart illustrating an embodiment of a process performed by a luma clipper;

Fig. 5 is a flow chart illustrating an embodiment of a process performed by the system of Fig. 1;

Fig. 6 is a data flow diagram illustrating an embodiment of the operation of the safe color limiter of Fig. 2;

Fig. 7 is a flow chart illustrating an embodiment of a process performed by the conversion term generator of Fig. 6;

Fig. 8 is a flow chart illustrating an embodiment of a process performed by the luma converter of Fig. 6;

Fig. 9 is a flow chart illustrating an embodiment of a process performed by the chroma converter of Fig. 6;

Fig. 10 is a flow chart illustrating an embodiment of a process performed by the chroma clipper of Fig. 6;

Fig. 11 is a flow chart illustrating an embodiment of a process performed by the safe color limiter of Fig. 6;

Fig. 12 is a flow chart illustrating an embodiment of a process performed by the correction generator of Fig. 6;

Fig. 13 is a flow chart illustrating an embodiment of a process performed by the safe color generator of Fig. 6;

Fig. 14 is a block diagram illustrating a known format for a color component signal;

Fig. 15 is a data flow diagram illustrating an embodiment of a safe odd luma generator;

Fig. 16 is a flow chart illustrating an embodiment of a process performed by the safe odd luma generator of Fig. 15;

Fig. 17A is an illustration of the double cone space color space of a Y_{601} C_b C_r component color;

Fig. 17B is a cross-sectional view of the double cone of Fig. 17A;

Fig. 18A is a diagram illustrating the double cone representing safe colors of a YUV_{IRE}, component color;

Fig. 18B is a cross-sectional view of the double cone of Fig. 18A;

Fig. 19 is an illustration of a double triangle representing the double cone of Fig. 18A collapsed into two-dimensional space;

Fig. 20 is the double triangle of Fig. 19 in safe space;

Fig. 21 is a diagram illustrating how space color modification is applied to unsafe pixels; and

Fig. 22 is a diagram illustrating how the safe color modification applied in Fig. 21 affect the two chroma components of the component color.

DETAILED DESCRIPTION

The following detailed description should be read in conjunction with the attached drawing in which similar reference numbers indicate similar structures. All references cited herein are hereby expressly incorporated by reference.

Safe color modification modifies a luma component and chroma components of a color to produce the closest safe color of a three-dimensional color space. Safe color modification simplifies the process of modifying the color components by transforming each component of a pixel into a coordinate system, referred to herein as safe color space or safe space. After the pixel values are transformed into safe space, safe color modification checks whether a combination of the transformed values is unsafe. If the combination is unsafe, the correction values that are applied to the original color components are determined, and then applied to the original color components. The transformation into safe space allows simpler mathematical operations to be performed in determining the correction values and applying these values to the components. The simpler operations allow the color corrections to be applied to the luma and chroma

components at a faster rate, thus increasing the rate at which safe color modification of a color can be performed.

The safe color transformation creates symmetrical relationships between the components of a component color that simplify the mathematical operations and provide a few common formulas capable of being applied to determine all correction values, regardless of the value of unsafe color to be corrected. The symmetrical relationships are realized by incorporating three general steps into the transformation: a) converting the elliptical cross-section of a three-dimensional safe color double-cone of a three-dimensional space into a circular cross-section, b) collapsing the three-dimensional double-cone into a two-dimensional double-triangle in two-dimensional color space, and c) centering a luma axis defined by the double-triangle about the luma origin such that the double triangle is collapsed into a triangle defined by only positive values. Described below is a system and method of safe color modification including a transformation into safe space that incorporates the three general steps. Following the description of system and method is a detailed description of the principles from which the system and method are derived.

FIG. 1 is a data flow diagram illustrating an embodiment of a safe color modification system 1. A color limiter 4 has an input for receiving a component signal 2 and an input for receiving color limiting parameters 8. Applying the color limiting parameters 8 to the component signal 2, the color limiter 4 generates a safe component signal 6 through its output.

FIG. 2 is a data flow diagram showing an embodiment of the safe color modification system 1 of FIG. 1 in more detail. The component signal 2 includes a first chroma component 10, a second chroma component 12, and a luma component 14. The color limiter 4 includes a luma clipper 16 and a safe color limiter 24. The safe component signal 6 includes a safe first chroma component 26, a safe second chroma component 28, and safe luma component 30. The color limiting parameters 8 include luma threshold values 20 and safe color parameters 22.

The luma clipper 16 receives through its inputs a luma component 14 and luma threshold values 20. The luma clipper 16 then generates a checked luma component 18 that is sent through its output to the safe color limiter 24 in a manner described in more detail below in connection with Fig. 4. The safe color limiter 24 receives through its inputs components 92 and the safe color parameters 22. The components 92 include the

first chroma component 10, the second chroma component 12, and the checked luma component 18. The safe color limiter 24 generates the safe first chroma component 26, the safe second chroma component 28, and the safe luma component 30, in a manner to be described below in connection with Fig. 5.

In an embodiment of safe color modification, the luma threshold values are converted from IRE units as defined by NTSC into $Y'_{601}C_rC_b$ (601 units). This embodiment may be used, for example, if the luma threshold values are received from a user interface (not shown) that operates with IRE units, and the received component signal 2 is in 601 units. An example of such a user interface is described in U.S. Patent Application "Source Color Modification on a Digital Nonlinear Editing System" by Robert Gonsalvez, filed on even date herewith, and in "Apparatus and Method for Generating and Displaying Histograms of Color Images for Color Correction," by Robert Gonsalvez, filed on even date herewith.

FIG. 3 is a data flow diagram illustrating an embodiment of luma threshold conversion. A luma threshold converter 36 receives through its inputs a luma high threshold value 32 in IRE units, a luma low threshold value 34 in IRE units, and a black level setup value 45. The black level setup value is a value representing an offset that is applied to a luma value in IRE units before converting to a luma component Y of a component signal in 601 units. The luma threshold converter 45 applies the black level setup value 45 to the IRE threshold values 32 and 34 to generate a luma high threshold value 40 in 601 units and a luma low threshold value 38 in 601 units at its output. The luma high and luma low threshold values 40 and 38, respectively, which are the luma threshold values 20 of FIG. 2, may be generated by the execution of the following equations.

$$Equation\ 1a:\ Y\_Clip\_High_{601} = \frac{219 \bullet (Y\_Clip\_High_{IRE} - setup)}{(100 - setup)} + 16,$$

$$Equation\ 1a:\ Y\_Clip\_Low_{601} = \frac{219 \bullet (Y\_Clip\_Low_{IRE} - setup)}{(100 - setup)} + 16,$$

where $Y\_Clip\_High_{601}$ is the luma high threshold value 40 in 601 units, $Y\_Clip\_High_{IRE}$ is the luma high threshold value 32 in IRE units, setup is the black level setup value 45, $Y\_Clip\_Low_{601}$ is the luma low threshold value 38 in 601 units, and $Y\_Clip\_Low_{IRE}$ is

the luma low threshold value 34 in IRE units. The derivation of Equations 1a and 1b is described in detail below.

FIG. 4 is a flow chart illustrating the operation of the luma clipper (16 in Fig. 2, 42 in Fig. 3). In step 110, a luma high threshold value 40 is compared to the value of the luma component 14. If it is determined that the value of the luma component 14 is greater than the luma high threshold value 40, then the value of the luma component is clipped to the luma high threshold value 40, i.e., the checked luma component 18 is set equal to the luma high threshold value 40 in step 112. In an embodiment, an interrupt or event may be generated in step 113 to indicate that a luma value is too high. This interrupt or event may be used to interrupt the safe color modification process in order to inform a user that a luma component with a value outside of the range defined by the threshold values has been received. In one embodiment, the interrupt or event may be addressed after all the pixels of a display field have been processed by the color limiter 4. In an alternative embodiment, no interrupt or event is generated in step 113, and the luma clipper 16 automatically clips the luma component 14.

If the value of the luma component 14 is not greater than the luma high threshold value, then in step 114 the luma component is not clipped, i.e., the checked luma component 18 remains equal to the value of the luma component 14. In step 116, the checked luma component 18 is then compared to the luma low threshold value 38. If the checked luma component 18 is less than the luma threshold value 38, then the luma component 14 is clipped to the luma low threshold value, i.e., the checked luma component 18 is set equal to the low threshold value in step 118. In step 119, an interrupt or event may be generated, similar to step 113. If in step 116 it is determined that the checked luma component 18 is not less than the luma threshold value, then the luma component is not clipped, i.e., in step 120 the checked luma component remains equal to the luma component 14. Comparisons of the luma component to the high and low thresholds may be performed in any order or in parallel.

FIG. 5 is a flow chart illustrating the operation of the safe color limiter 24 of FIG. 2. In step 122, conversion terms are generated from the safe color parameters 22. In step 124, the conversion terms are applied to the safe color limiter inputs 92 to generate converted color components, which include a converted luma component, a converted first luma component, and a converted second chroma component. In step 128, it is determined whether a combination of the converted luma and chroma

components exceeds a threshold value. If the combination of components does not exceed the predetermined threshold value in step 128, the components are output as the safe color components in step 130. Otherwise, if the combination does exceed a threshold value in step 128, then in step 132 color correction terms are generated. In step 134, the color correction terms are applied to the safe color limiter inputs. After the color correction terms are applied to the color component signal in step 134, the safe color components of the components 6 (Fig. 1) then may be combined to form a safe composite signal that can be used for broadcast or transmission.

FIG. 6 is a data flow diagram illustrating the operation of the safe color limiter 24 of FIG. 2 in more detail. The safe color limiter 24 includes a conversion term generator 44, a color converter 58, a safe color checker 74, a correction generator 80, and a safe color generator 90. The color converter 58 includes a luma converter 60, a chroma clipper 62, and a chroma converter 64. The safe color parameters 22 include a color high threshold value 49, a color low threshold value 47, and the black level setup value 45. The color threshold values 49 and 47 define a high and low threshold value, respectively, for a combination of the checked luma component 18, the first chroma component 10, and the second chroma component 12.

The conversion term generator 44 receives the color high threshold value 49, the color low threshold value 47, and the black level setup value 45. The conversion term generator 44 generates conversion terms 46, which include a luma scale value 48, a luma offset value 50, a first chroma scale value 52, a second chroma scale value 54, and an inverse luma scale value 56. The inverse luma scale value 56 is an inverse of the luma scale value 48.

The color converter 58 receives the conversion terms 46 and safe color limiter inputs 92, which include the checked luma component 18, the first chroma component 10, and the second chroma component 12. The color converter applies the conversion terms 46 to the color limiter inputs 92 to generate converted color values 66. The converted color values 66 include a converted luma value 68 and a converted chroma value 70. The luma converter 60 receives the luma scale value 48, the luma offset value 50, and the checked luma component 18 as inputs and generates the converted luma value 68 at an output. The chroma converter 64 receives the first chroma scale value 52, the second chroma scale value 54, the first chroma component 10, and the second chroma component 12 as inputs and generates a combined chroma value 72 at its output.

The chroma clipper 62 receives the combined chroma value 72 and the converted luma value 68 and generates the converted chroma value 70.

The safe color checker 74 receives the converted color values 66, generates correction flags 76, and may generate an interrupt 75. In some embodiments, an event may be generated. In an embodiment, the correction flags 76 are stored in an odd luma register 78, which is described in more detail below.

The correction generator 80 receives correction flags 76, the inverse luma scale value 56, the combined chroma values 72 and the converted color values 66. The correction generator 80 generates color correction values 82, which include a luma correction value 88 and a chroma correction value 84, to be applied to the safe color limiter inputs 92.

The safe color generator 90 receives the safe color limiter inputs 92 and the color correction values 82, and generates the safe component signal 6, which includes the safe luma component 26, the safe first chroma component 28, and the safe second chroma component 30. If no modification is to be made to inputs 92, the inputs 92 simply pass through the generator 90 as the output components 6.

FIG. 7 is a flow chart illustrating an embodiment of the operation of the conversion term generator 44. In steps 132-136, the color high threshold value 49, the color low threshold value 47, and the black level setup value 45, respectively, are received. These steps and other steps in this flowchart may be performed in any order or in parallel. In step 138, a luma scale factor 48 is generated. In an example embodiment where the component signal is a 601 signal, the luma scale factor 48 is generated by the execution of the following equation:

$$Equation\ 2:\ Y\_Scale = \frac{256 \bullet (100 - setup)}{219 \bullet (Color\_High - Color\_Low)},$$

where setup is the black level setup value 45 , Color_High is the color high threshold value 49, and Color_Low is the color low threshold value 47. The derivation of Equation 2 is described in detail below.

In step 140, the luma offset 50 is generated. In the example embodiment where the color component signal is a 601 signal, the luma offset 50 is generated by execution of the following equation:

$$Equation\ 3:\ Y\_Offset = \frac{256 \bullet (235\_setup - 1600 - 109.5 \bullet (Color\_High + Color\_Low)}{219 \bullet (Color\_High - Color\_Low)},$$

where setup is the black level setup value 45, Color_High is the color high threshold value 49, and Color_Low is the color low threshold value 47. The derivation of Equation 3 is described in detail below.

In step 142, the first chroma scale value 54 is generated. In the example embodiment where the color component signal is a 601 signal, the first chroma scale value is generated by the following equation:

$$Equation\ 4:\ C_B Scale = \frac{16 \bullet 0.436010346 \bullet (100 - setup)}{7 \bullet (Color\_High - Color\_Low)},$$

where setup is the black level setup value 45, Color_High is the color high threshold value 49, and Color_Low is the color low threshold value 47. The derivation of Equation 4 is described in detail below.

In step 144, the second chroma scale value 54 is generated. In the example embodiment where the color component signal is a 601 signal, the second chroma scale value is defined by the equation:

$$Equation\ 5:\ C_R Scale = \frac{16 \bullet 0.614975383 \bullet (100 - setup)}{7 \bullet (Color\_High - Color\_Low)},$$

where setup is the black level setup value 45, Color_High is the color high threshold value 49, and Color_Low is the color low threshold value 47. The derivation of Equation 5 is described in detail below.

In step 146, an inverse luma scale factor is generated. In the example embodiment where the color component signal is a 601 signal, the inverse luma scale factor is simply the inverse of Equation 2 above:

$$Equation\ 6:\ \frac{1}{Y\_Scale} = \frac{219 \bullet (Color\_High - Color\_Low)}{256 \bullet (100 - setup)},$$

FIG. 8 is a flow chart illustrating an embodiment a process performed by the luma converter 60. In steps 148 through 150, the luma scale value 48, the luma offset value 50, and the checked luma component 18 are received. The order of steps 148 through 150 may be performed in any order or in parallel.

In step 152, the converted luma value 68 is generated. In the example embodiment where the color component signal is a 601 signal, the converted luma value may be defined by the following equation:

$$Equation\ 7: \quad Y_{SS} = Y_{CH} \bullet Y\_Scale + Y\_Offset,$$

where Y_{SS} is the converted luma value 68 and Y_{CH} is the checked luma component 18, Y_Scale is the luma scale value 48, and Y_Offset is the luma offset value 50. The derivation of Equation 7 is described in detail below.

Fig. 9 is a flow chart illustrating an embodiment of a process performed by the chroma converter 64. In steps 153 and 155, the first and second chroma scale values 52 and 54, and the first and second chroma components 10 and 12 are received.

In step 157, a converted first chroma component is generated. In the example embodiment where the color component signal is a 601 signal, the converted first chroma component may be defined by the following equation:

$$\text{Equation 8:} \quad U_{SS} = (C_B - 128) \bullet C_B Scale,$$

where U_{SS} is the converted first chroma component, C_B is the first chroma component 10, and C_B_scale is the first chroma scale value 52. The derivation of Equation 8 is described in detail below.

In step 159, a converted second chroma component is also generated. In the example embodiment, if a color component signal is a 601 signal, the converted second chroma component is defined by equation 9 as follows:

$$Equation\ 9: \quad V_{SS} = (C_R - 128) \bullet C_R Scale,$$

where V_{SS} is the converted second chroma component, C_r is the second chroma component 12, and C_r_Scale is the second chroma scale value 54. The derivation of equation 9 is described in detail below.

In step 156, the combined chroma value 72 is generated by combining the converted first and second chroma components. In the example embodiment where the color component signals a 601 signal, the combined chroma value 72 is defined by equation 10 as follows:

Equation 10: $C_{SS} = \sqrt{U_{SS}^2 + V_{SS}^2}$,

where C_{SS} is the combined chroma value 72, U_{SS} is the converted first chroma component, and V_{SS} is the converted second component. The derivation of equation 10 is described in detail below.

Thus, equations 7-10 implement the transformation into a safe color space that permit simpler operations to be performed on color components. Such operations may include checking whether the color is safe and if the color is not safe, and determining the corrections to be applied to the components, as described in detail below.

Fig. 10 is a flow chart illustrating an embodiment of a process performed by the chroma clipper 62. In steps 160 and 161, the converted luma value 68 and a combined chroma value 72 are received. The steps 160 and 161 may be performed in any order or in parallel. In step 162, the converted luma value 68 is combined with a predetermined threshold value to produce a chroma threshold value. In the example embodiment, where the color component signal is a 601 signal, the chroma threshold value is defined by equation 11 as follows:

Equation 11: $|Y_{SS}| + 128$,

where Y_{SS} is the converted luma value 68. The derivation of equation 11 is described in detail below.

In step 163, the combined chroma value 72 is compared to the chroma threshold value. If it is determined in step 163 that the combined chroma value 72 is greater than the chroma threshold value, then at step 165, the combined chroma value 72 is clipped to the chroma threshold value to produce the converted chroma value 70. If it is determined in step 163 that the combined chroma value 70 is not greater than the chroma threshold value, then in step 164 the converted chroma value 70 is set equal to the combined chroma value.

FIG. 11 is a flow chart illustrating an embodiment of a process performed by the safe color checker 74 of Fig. 6. In step 166, the converted luma value 68 and the converted chroma value 70 are received. In step 168, the magnitude of the converted luma value 68 and the converted chroma value 70 are combined to generate a combined value. In step 170, the combined value is compared to a predetermined threshold value. If it is determined in step 172 that the combined value is not greater than the

predetermined threshold value, then a correction flag is cleared in step 174. In an alternative embodiment, the correction flag is initialized between steps 170 and 172, and in the event that the combined value is not greater than the predetermined threshold value, the correction flag remains cleared.

If in step 172 it is determined that the combined value is greater than the predetermined threshold value, then in step 176 it is determined whether the converted luma value 68 is a positive or negative value. If it is determined that the converted luma value 68 is a positive value, then in step 177 a "color too high" correction flag is set and in step 178 a "color too high" interrupt is generated. In an alternative embodiment, a "color too high" event is generated in step 178. If the converted luma value is a negative value, then in step 179 a "color too low" correction flag is set and in step 180 a "color too low" interrupt is generated. In an alternative embodiment, a "color too low" event is generated on step 180.

The correction flags are used in order to determine the color correction values to be generated by the color correction generator 80. The interrupts, or alternatively the events, are used to inform the system that an unsafe color has been detected. In an embodiment, after the entire field of which the current pixel belongs has been processed, the color too high and color too low interrupt or events either may be ignored or may be used to generate a warning for a system user. In an alternative embodiment, interrupts or events are generated after each pixel or each line of a video field has been processed. In yet another alternative embodiment, no interrupt or event is generated, and the safe color modification is automatically executed.

FIG. 12 is a flow chart illustrating an embodiment of a process performed by the correction generator 80. In steps 182-188, the combined chroma value 72, the converted luma value 68, the converted chroma value 70, the color correction flags 76, and the inverse luma scale value 56 are received. The steps 182-188 may be performed in any order or in parallel. In step 190, the color correction generator 80 determines whether either of the correction flags 76, the "color too high" flag or the "color too low" flag, are set. If neither of the correction flags 76 are set, a luma correction offset 88 is set to zero in step 192 and a chroma correction scalar 84 is set to one in step 194. The values for the luma correction offset 88 and the chroma correction scalar are called "passthrough" values because these "passthrough" values allow the original luma and chroma

components, 18, 10, and 12, respectively, to pass through the safe color generator 90 unmodified.

If either of the correction flags are set, a luma correction offset is generated in step 196. In the example embodiment where the color component signal is a 601 signal, the luma correction offset is defined by the following equation:

$$Equation\ 12: \quad Y_C = \frac{|Y_{SS}| - 128 + C_{CH}}{2} \bullet Inverse\_Y\_Scale,$$

where Y_c is the luma correction offset 88, Y_{SS} is the converted luma value 68, C_{CH} is the converted chroma value 70, and inverse_Y_scale is the inverse luma scale value 56. If the color too high correction flag is set, then the luma correction offset 88 will have a positive value. If the color too low correction flag was not set or cleared, then the luma correction offset 88 has a negative value. The derivation of Equation 12 is described in more detail below.

In step 198, the chroma correction scalar 84 is generated. In the example embodiment where the color component signal is a 601 signal, the chroma correction scaler 84 is defined by the following equation:

$$Equation\ 13: \qquad C_C = \frac{128 - |Y_{SS}| + C_{CH}}{2 \bullet C_{SS}},$$

where C_C is the chroma correction scalar 84, Y_{SS} is the converted luma value 68, C_{CH} is the converted chroma value 70, and C_{SS} is the combined chroma value 72.

FIG. 13 is a flow chart illustrating an embodiment of a process performed by the safe color generator 90 of Fig. 6. In steps 202 and 204, the safe color generator 90 receives the luma correction offset 88 and the chroma correction scaler 84. The steps 202 and 204 may be performed in any order or in parallel. In step 208, the luma correction offset 88 is added to the checked luma component 18 to generate the safe luma component 26.

In steps 209-211, the first chroma component 10 is offset by a chroma offset value, multiplied by the first chroma correction scalar 84, and offset again by an opposite value of the chroma offset value to generate the first safe chroma component 28. In the example embodiment where the component color signal is a 601 signal, the safe first chroma component 28 is defined by the following equation:

$$Equation\ 14: \quad C_{Bcorr} = [\,(C_B - 128) \bullet C_C\,] + 128,$$

where C_{corr} is the safe first chroma component, C_B is the first chroma component 10, C_C is the chroma correction scalar value 84, and 128 is the chroma offset value.

In steps 212, 213, and 215, the second chroma component is offset by the chroma offset value, multiplied by the chroma correction scalar 84, and then offset by the opposite value of the chroma offset value to generate the second safe chroma component 30. In the example embodiment where the color component signal is a 601 signal, the safe second chroma component 30 is defined by following equation:

$$Equation\ 15: \quad C_{Rcorr} = [\,(C_R - 128) \bullet C_C\,] + 128,$$

where C_{Rcorr} is the second safe chroma component 30, C_R is the second chroma component 12, C_C is the chroma correction value 84, and 128 is the chroma offset value.

In one embodiment, the color limiting parameters 8 are constrained. In particular, the luma high threshold value 40 is less than or equal to the color high threshold value 49, and the luma low threshold value 38 is greater than or equal to the color low threshold value 47. The black level setup value 45 also ranges from 0 to 7.5 IRE. The color low threshold value also ranges from -40-10 IRE, and the color high threshold value ranges from 90-133 IRE.

In the example embodiment, if the color component signal is a 601 signal, the luma high threshold value is 255 and the luma low threshold value is 0, because the luma component for a 601 is within that range. Using the luma high threshold and luma low threshold values 40 and 38 in combination with the assumed values above, for a 601 signal, the conversion terms are defined as follows: the luma scale offset 48 ranges from 0.625-1.461; the luma offset value 50 ranges from -183.379-47.770; the first chroma scale value 52 ranges from 0.5329-1.2457; the second chroma scale value 54 ranges from 0.7516-1.7571; and the inverse luma scale value 56 ranges from 0.684-1.600.

Some component color formats, such as the 601 format do not include chroma components to represent every pixel of a digital image, because a 601 signal samples the luma component at twice the frequency that it samples the chroma components. Thus, some pixels are represented by a luma component and chroma components, and other pixels are represented by only a luma component.

FIG. 14 is a data flow diagram illustrating a 4:2:2 Y_{601} C_BC_R component signal. C_{BO}, Y_O and CR_O represent an even pixel 214, having a luma component Y_O and two chroma components C_O and CR_O. Y_I represents an odd pixel 216 represented only by a luma component, and pixel 218 represents another even pixel. In one embodiment of safe color modification, the chroma components of even pixels 214 and 218 are interpolated to generate the chroma components of the odd pixel 216, and then safe color modification is applied to those values. Since this embodiment may use significant processing resources, an alternative embodiment of safe color modification for odd luma components that requires fewer resources is illustrated in Fig. 15 and Fig. 16.

FIG. 15 is a data flow diagram illustrating safe color modification of an odd luma component. The luma clipper 16 receives an odd luma component 16 and generates a checked odd luma component 228. This operation is analogous to how a checked luma component 18 is generated from a luma component 14, as described above with respect to Figs. 2-4. The safe odd luma generator 230 receives the checked odd luma component 228, a preceding safe luma component 220, a preceding color correction flags 222, a next safe luma component 224, and a next color correction flags 226, and generates a safe odd luma component 232.

Referring to FIG. 14, for the odd luma component 216, the preceding safe luma component would be the safe luma component resulting from applying safe color modification to the preceding as described in FIGS. 1, 2 and 5. The preceding color correction flags are the resulting flag 76 of Fig. 5 resulting from applying safe color modification to the even pixel 214. The next safe luma component 224 is the safe luma component 26 in Fig. 5 resulting from applying safe color modification to the even pixel 218. The next color correction flags 226 are the color correction flags 76 in FIG. 6 resulting from applying safe color modification to the even pixel 218.

FIG. 16 is a flow chart illustrating safe color limiting of an odd luma component. In step 234, the luma clipper 16 receives the odd luma component 216, and in step 236, the odd luma component may be clipped in accordance with the process described in FIG. 4. In step 238, the preceding color correction flags are accessed from the odd luma register of FIG. 6. In step 240, it is determined whether the odd luma component is the last luma component of the display field. If the odd luma component is the last component of the display field, then the safe odd luma component 232 will be derived only from a previous even pixel. For example, in FIG. 14, if odd luma component 216 is

the last luma component of the display field, then the odd luma component 216 would only be concerned with the even pixel 214.

If the odd luma component is not the last component of the display field, the safe odd luma generated 230 accesses the next color correction flags 226 from the odd luma register 78. Next, the safe odd luma generator 230 determines whether either of the next color correction flags are set. If either of the next color correction flags are set, then in step 248 the preceding safe luma component is accessed, and in step 250 the next safe luma component is accessed. Next, in step 252, the safe odd luma component is set equal to the average of the preceding safe luma component and the next safe luma component.

If in step 244 it is determined that neither of the next color correction flags are set, then in step 246 it is determined whether either of the preceding color correction flags are set. If either of the preceding color correction flags are set, then in step 248 the preceding safe luma component is accessed, and in step 250 the next safe luma component is accessed. Finally, in step 252, the safe odd luma component 232 is set equal to the average of the preceding safe luma component in the next safe luma component. The steps of 244 may be performed in any order or in parallel.

If in step 246 it is determined that neither of the preceding color correction flags is set, then in step 256, the safe odd luma component 232 is set equal to the odd luma component 216.

If in step 240 it is determined that the odd luma component is the last luma component of the display field, then in step 254 it is determined whether either of the preceding color correction flags are set. If either of the preceding color correction flags are set, then in step 258, the preceding safe luma component is accessed, and in step 260, the safe odd luma component 232 is set equal to the preceding safe luma component.

If in step 254 it is determined that neither of the preceding color correction flags are set, then in step 256, safe odd luma component 232 is set equal to the odd luma component 216.

In one embodiment, the preceding color correction flags and the next color correction flags also may be the result of an OR operation on the preceding color correction flags and the next color correction flags, respectively.

Thus equations 7-10 transform the color components into safe space, where it is determined whether the color is safe. If it determined that the color is unsafe, equations

12 and 13 may be applied to determine the corrections to be applied to the color components to generate a safe color. The corrections then may be applied the color components.

Having described the system and method of safe color modification, the principles from which the system and method are derived will now be discussed.

Generally checking for a safe color of a pixel includes determining that the luma is within range of values, and determining that the sum of the luma and the chroma is within range of values. This checking may be summarized by the following equations: $Y \leq$ luma_high; $Y \geq$ luma_low; $Y + C \leq$ color value _high; and $Y - C \geq$ color_low, where the variable C is defined by $C = \sqrt{U^2 + V^2}$, Y is the luma and U and V are chroma components, color_high is a color high threshold value, color_low is a color low threshold value, luma_high is a luma high threshold value, and luma_low is a luma low threshold value. Luma_high is less than or equal to chroma_high, and luma_low is greater than or equal to chroma_low, as discussed in more detail below.

Throughout the description that follows, safe color modification is described with respect to a 4:2:2 $Y_{601}C_bC_r$ video format (601 format). The principles of safe color modification may be applied to other video formats as well.

In order to test for the color limits, the 601 components are converted to YUV space. This format is commonly used to convert video components of a component signal into a composite signal.

However, the luma_high and luma_low values, and the color_high and color_low values may be expressed in IRE units. Thus, after converting into YUV format, the components are converted to IRE units. To simplify the analysis, and the calculation of modifications to be applied to components, the YUV_{IRE} components are then transformed into a "safe space" coordinate system. The mathematical operations for converting from 601 format to YUV format to YUV_{IRE} format to safe space are described below. These operations are then combined for transforming from 601 format directly to safe space. The resulting transformations are embodied in three equations, each equation specific to a color component. Each equation is applied to its corresponding component to convert the component from 601 format directly to safe space.

After determining the transformations from 601 format to safe space, modifications determined in safe space can be converted from safe space to 601 format by applying inverse transformations to the modifications. These modifications can then be applied to the original color components in 601 format.

The following description considers the even pixels of a 4:2:2 signal. As discussed above, the odd luma components of a 4:2:2 signal are handled differently and in the manner shown in Figs. 14-16. To convert from YC_BC_R to IRE, it is easier to first convert to YUV space where Y has a range from 0 to 1, and Y_{601} has a range of 16 - 235, leading to the conversion equation:

$$\textit{Equation 16 :} \qquad Y = \frac{1}{219} \bullet (Y_{601} - 16)$$

where, Y_{601} represents the luma component or Y component in the 601 format, and Y is the luma component of the YUV format.

The C_B and the C_R components have a range from 16 - 240 with an offset of 128. After the offset is removed, they cover a range of plus or minus 112. The U chroma component has a range of ± 0.436010346 and the V chroma component has a range of ± 0.614975383. These ranges may be derived from the definitions of the YUV and YC_BC_R formats described herein. For a more detailed description of the color video and color science describing the derivation of these conversion values, see Charles Poynton's Color FAQ: http://www.inforamp.net/-poynton. Thus, the following equations 17 and 18, for converting the chroma components in 601 format to YUV format are derived:

$$\textit{Equation 17 :} \qquad U = \frac{0.436010346}{112} \bullet (C_B - 128)$$

$$\textit{Equation 18 :} \qquad V = \frac{0.614975383}{112} \bullet (C_R - 128)$$

Converting to YUV space creates the correct ratios between Y, U and V for encoding to a composite signal. To actually convert to an IRE scale for a real composite signal, the black level set up as defined for NTSC also is used. Thus, for converting from YUV to YUV_{IRE} the following equations 19 , 20 and 21 are used:

$$\textit{Equation 19:} \quad Y_{IRE} = Y \bullet (100 - setup) + setup$$

$$\textit{Equation 20:} \quad U_{IRE} = U \bullet (100 - setup)$$

$$\textit{Equation 21:} \quad V_{IRE} = V \bullet (100 - setup)$$

where setup and100 define the values black and white, respectively, of a composite signal.

Combining equations 16 through 21 results in the following equations 22, 23, and 24 to convert directly from Y C_BC_R to YUV_{IRE}. Equation 22, 23 and 24 are defined as follows:

$$\textit{Equation 22:} \quad Y_{IRE} = \frac{1}{219} \bullet (100 - setup) \bullet (Y_{601} - 16) + setup$$

$$\textit{Equation 23:} \quad U_{IRE} = \frac{0.436010346}{112} \bullet (100 - setup) \bullet (C_B - 128)$$

$$\textit{Equation 24:} \quad V_{IRE} = \frac{0.614975383}{112} \bullet (100 - setup) \bullet (C_R - 128)$$

Using Equation 22, the formula to convert from Y_{IRE} back to Y_{601} is derived, which is used for converting luma threshold values from IRE units to 601 units as described above with reference to Fig. 3. The equation for converting for Y_{IRE} back to Y_{601} is defined in equation 25 below:

$$\text{Equation 25:} \quad Y_{601} = \frac{219 \_ (Y_{IRE} - setup)}{(100 - setup)} + 16$$

For three-component color formats such as the 601 format, where one component is a luma component and the two other components are chroma components, the 3D representation of the valid color region looks like a double cone centered on the luma axis as illustrated in FIG. 17A for the 601 format. It should be noted that because the 601 format has an offset of 128 for both the C_B and the C_R component, the double cone 270 is offset from the luma axis Y_{601}. FIG. 17B illustrates a view looking down the axis of the double cone from point A of FIG. 17A. FIG. 17B illustrates that the cross-section 272 of the valid color region for Y C_BC_R has an elliptical shape.

FIG. 18A illustrates the valid color region 274 in YUV_{IRE} color space. Applying Equations 22, 23 and 24 to the double-cone valid color region 270 of FIG. 17A produces the double cone 274 of FIG. 18A centered around the Y_{IRE} axis.

FIG. 18B is a view of the double cone looking down the Y_{IRE} axis and shows that after applying Equations 22, 23 and 24, the cross-section of the double cone 276 of the YUV_{IRE} color space is a circle. Because the three components of the YC_BC_R color space have been normalized by Equations 22, 23, 24 into the YUV_{IRE} color space, a unit excursion in any dimension of the YUV_{IRE} color space of FIG. 18A is of equal significance. The slope of the cone edges is exactly 45°. Thus, the application of equations 22-24 to the 601 format to produce components in the YUV_{IRE} format implements the first general step of the transformation. The significance of the symmetrical properties of the double cone of YUV_{IRE} space is described in more detail below.

First, since the double cone 274 is circular, the magnitude of the combined chroma components, U_{IRE} and V_{IRE}, can be calculated, and the 3D cone can be collapsed into a 2D triangle.

FIG. 19 illustrates the collapsing color space. This collapse or conversion is achieved by converting the U_{IRE} and V_{IRE} components into a C_{IRE} component defined by

$$C_{IRE} = \sqrt{U_{IRE}^2 + V_{IRE}^2},$$

In FIG. 19, lines 282 and 284 represent the safe color limits in the two-dimensional YC_{IRE} space.

In order implement the third general step of the transformation so that the equations for safe color modification are simpler, the next step is to shift the double triangle 280 downward along the Y_{IRE} axis so that it is centered about the C_{IRE} axis, and scaling the safe color limit 282 to a predetermined threshold value in safe color space. Figure 20 illustrates the safe color double-triangle 281 in safe color space. In an embodiment of safe color modification, the number 128 is used as the predetermined threshold value. The number 128 is used because it is a convenient number for performing mathematical operations. 8 bits, e.g., 7 magnitude bits and a sign bit, can be used to represent values ranging from -128 to 128, which is the same number of bits used to represent a color component in 601 format. In a hardware implementation of safe

color modification, this symmetry between the number of bits leads to simpler and consequently faster circuitry.

The shifting and scaling involves converting coordinates of the YC_{IRE} two-dimensional coordinate space into the $Y C_{SS}$ safe color space. In order to make this conversion from $Y C_{IRE}$ color space to safe color space, the three original components of the color component signal, Y_{601}, C_B, and C_R, are scaled by an additional factor. An offset also is applied to the original luma component Y_{601}. Equations 26, 27 and 28 define the conversion from the 601 components to safe color component space:

$$Equation\ 26: \quad Y_{SS} = Y_{IRE} \bullet scale_Y + offset$$

Known values of points 900, 902, 904, 900', 902', and 904' from FIG. 19 are applied to provide

$$Equation\ 27: \quad U_{SS} = U_{IRE} \bullet scale_C$$

$$Equation\ 28: \quad V_{SS} = V_{IRE} \bullet scale_C$$

In FIG. 20, the following equations 29, 30 and 31 defining $scale_Y$, offset, and $scale_C$ are obtained:

$$Equation\ 29: \quad scale_Y = \frac{256}{Color\_High - Color\_Low}$$

$$Equation\ 30: \quad offset = 128 - Color\_High \bullet \frac{256}{Color\_High - Color\_Low}$$

$$Equation\ 31: \quad scale_C = \frac{256}{Color\_High - Color\_Low}$$

Substituting equations 29, 30 and 31 back into Equations 26 through 28, the following equations for converting the components of a $Y C_B C_R$ signal into safe color space are defined:

Equation 32:

$$Y_{SS} = \frac{256\_(100 - setup)}{219\_(Color\_High - Color\_Low)} \_ Y'_{601} + \frac{256\_(235\_setup - 1600 - 109.5\_(Color\_High - Color\_Low))}{219\_(Color\_High - Color\_Low)}$$

$$Equation\ 33: \quad U_{SS} = \frac{16 \bullet 0.436010346 \bullet (100 - setup)}{7 \bullet (Color\_High - Color\_Low)} \bullet (C_B - 128)$$

$$Equation\ 34: \quad V_{SS} = \frac{16 \bullet 0.614975383 \bullet (100 - setup)}{7 \bullet (Color\_High - Color\_Low)} \bullet (C_R - 128)$$

From Equations 32, 33 and 34, Equations 2 - 6 are derived.

FIG. 21 is a graph illustrating the next step of the conversion process of taking the magnitude of the safe space luma component Y_{SS}, and using the limit line 286 that defines the line the magnitude of Y + C = 128 to determine whether a color is safe. If the sum of the magnitude of luma of a pixel in safe space, Y_{SS}, and the chroma of the pixel in safe space, C_{SS} is greater than 128, then the color of the pixel is not safe. If the sum of the magnitude of Y_{SS}, $1Y_{SS}1$, and C_{SS} is less than or equal to 128, then the color of the pixel is safe.

Referring to FIG. 21, if a pixel is outside the safe area 294, the luma and chroma are adjusted to bring the pixel to the safe limit 286. Simply reducing either the luma or the chroma individually may generates visually undesirable results. Better results can be obtained by changing both luma and chroma equally, providing the closest safe color to the original color of a pixel. Thus, the adjustment is performed by mapping the color of the unsafe pixel to the limiting line 286 in a direction perpendicular to the limiting line 286. This is illustrated in FIG. 21 for point D which represents the color of a pixel in safe color space. In order to correct the color of the pixel, the luma and chroma components of the pixel are adjusted along the line 296.

It is possible that the C_{SS} value of a pixel may be greater than 128, but Y_{SS} of the pixel is less than 128 so long as the luma_high threshold value is less than or equal to the color_high threshold value and the luma_low threshold value is greater than or equal to the color_low threshold value. These conditions are satisfied because the original Y_{601} value has been clipped as described above with reference to FIGS. 2 and 4, before entering the safe color limiter.

Points A and B in FIG. 21 illustrate two situations where the C_{SS} component of the color of a pixel in safe space is greater than 128. With regard to point A, the same method of color modification discussed above with respect to point D may be applied to reduce the luma and chroma of the pixel by the same amount, thus mapping the point A down to point C_2 on the limit line 286 along the line 298.

However, if point B is mapped to the nearest point on the limit line 286, the resulting point is C_4, which is still an invalid color.

One solution to the problem posed by point B is to map all points that have a C_{SS} component greater than 128 back to the line 290, which defines C_{SS} =128, and then from line 290 perpendicularly to the nearest point on limit line 286. However, both points A and B are mapped to the same point C1. This mapping results in A and B having the same chroma value, although original point B had a greater chroma value than point B.

Alternatively, any points, such as B, that are to the right of line 288, having a chroma greater than the sum of the luma value and the threshold value may be mapped along path 312 and 310, and map points on the left of line 288 perpendicularly to the limit line 286. However, after this mapping, the chroma of point B is less than the chroma of point A, and the luma of point B is higher than point A.

Alternatively, the line 288 which defines $C_{SS} = |Y_{SS}| + 128$ may be used. For points to the left of this dashed line, $C_{SS} < Y_{SS} + 128$ and for points to the right of this dashed line, $C_{SS} > |Y_{SS}| + 128$. For example, point A is to the left of the line 288 and point B is to the right of the line 288. Using this mapping corrects all points to the left of line 288 in the manner described with respect to point D, by mapping the point perpendicularly to the line 286. In contrast, for points to the right of the line 288, using point B as an example, safe color limiting maps each point to the line 288, and then perpendicularly along the line 288 to the point C3. Point C3 represents a color in safe color space $|Y_{SS}| = 0$ and $C_{SS} = 128$. The chroma clipper 62 of FIG. 6 implements this type of chroma clipping as described in FIG. 18, steps 160 -166. The value of the converted chroma value 70 can be described by the following equation 35:

$$\text{Equation 35:} \quad C_{CH} = \begin{cases} |Y_{SS}| + 128 & \text{if} \quad C_{SS} > |Y_{SS}| + 128 \\ C_{SS} & \text{if} \quad C_{SS} \leq |Y_{SS}| + 128 \end{cases}$$

where C_{CH} is converted chroma value 70, Y_{SS} is the converted luma value 68, and C_{SS} equals the combined chroma value 72.

The point D in FIG. 21 is defined by coordinates ($|Y_{SS}|$, C_{CH}) and the new value to which pixels are limited is defined by the coordinates (Y_P, C_P). The results of changing just one of the components individually can be seen at points ($|Y_{SS}|$, C_I) and (Y_i, C_{CH}). Because the slope of the limit line 286 is -1, $C_i = 128 \ll Y_{SS} \ll$. Consequently, $Z = C_{CH} - C_i = C_{CH} - 128 + |Y_{SS}|$, where Z is the modification that is applied to a color to map it to closest safe color. The luma intercepts the limit line 286 at Y_P, where:

Equation 36 : $$Y_P = |Y_{SS}| - \frac{z}{2} = \frac{(|Y_{SS}| + 128 - C_{CH})}{2}$$

Solving for C_P:

Equation 37 : $$C_P = C_{CH} - \frac{z}{2} = \frac{(-|Y_{SS}| + 128 + C_{Checked})}{2}$$

After Y_P has been calculated, the correction factor Y_C to be applied to the original Y_{601} data to generate a safe luma component can be determined. This correction factor is scaled by the inverse of the scale factor that was originally applied to the Y_{601} data. Accordingly, Y_C can be defined by the following equation 38:

Equation 38 : $$Y_C = (|Y_{SS}| - Y_P) \bullet \frac{1}{Y_{scale}}$$

Equation 12 can be derived by combining Equations 36 and 38. The Y_C of Equation 38 is the luma correction value 88 of FIG. 6. As described above with reference to FIG. 6, the luma correction value 88 is then added to the checked luma component 18 to generate the safe luma component 26.

By adding the luma correction offset 88, as opposed to applying a scalar, to the checked luma component 18, round-off errors are minimized and pass-through, where no modifications are made is simplified by simply adding 0 to the checked luma component L8. Because the modification performed on the luma components is additive, the sign of the correction is pertinent, as described above with reference to FIG. 11, steps 176 - 180.

Unlike the checked luma component 18 which is modified with a luma offset value, the first chroma component 10 and the second chroma component 12 are modified using scale factors based on C_P . Figure 22 illustrates safe color space expanded into three-dimensional space to show the chroma components or vectors V_{SS}, U_{SS}, V_P, and U_P corresponding to chroma components C_{SS} and C_P, respectively. FIG. 22 is a view looking down the Y_{SS} axis at the circular chroma cross-section 314 of the safe color cone defining the safe color limit. Although the C_P vector representing the modified chroma component has the same angle as the C_{SS} vector, presenting the original chroma component, the magnitude of C_P is less than the magnitude of C_{SS}. The ratio of C_P to C_{SS} is the same as the ratio of U_P and V_P to U_{SS} and V_{SS}, respectively. U_P and V_P thus can be defined by the following equations:

Equation 40 : $$V_P = V_{SS} \bullet \frac{C_P}{C_{SS}}$$

Equation 39 : $$U_P = U_{SS} \bullet \frac{C_P}{C_{SS}}$$

Applying this same ratio to the original chroma components, CR and CB, and accounting for the 128 offset for chroma components of the 601 format, the safe chroma components may be defined by the following equations:

Equation 41 : $$C_{Bcorr} = [\,(C_B - 128) \bullet \frac{C_P}{C_{SS}}\,] + 128$$

Equation 42 : $$C_{Rcorr} = [\,(C_R - 128) \bullet \frac{C_P}{C_{SS}}\,] + 128$$

Equations 41 and 42 can be combined with equations 32-35 and 37 to derive equations 13-15, which define the chroma correction scalar and how it is applied to the original chroma components to produce safe chroma components.

Safe color modification may be implemented using a computer system. It should be understood that safe color modification is not limited by the specific computer described herein. Many other different machines may be used to implement the invention. Such a suitable computer system includes a processing unit which performs a variety of functions and a manner well-known in the art in response to instructions provided from an application program. The processing unit functions according to a program known as the operating system, of which many types are known in the art. The steps of an application program are typically provided in random access memory (RAM) in machine-readable form because programs are typically stored on a non-volatile memory, such as a hard disk or floppy disk. When a user selects an application program, it is loaded from the hard disk to the RAM, and the processing unit proceeds through the sequence of instructions of the application program.

The computer system also includes a user input/output (I/O) interface. The user interface typically includes a display apparatus (not shown), such as a cathode-ray-tube (CRT) display in an input device (not shown), such as a keyboard or mouse. A variety of other known input and output devices may be used, such as speech generation and recognition units, audio output devices, etc.

The computer system also includes a video and audio data I/O subsystem. Such a subsystem is well-known in the art and the present invention is not limited to the specific subsystem described herein. The audio portion of subsystem includes an analog-to-digital (A/D) converter (not shown), which receives analog audio information and

converts it to digital information. The digital information may be compressed using known compression systems, for storage on the hard disk to use at another time. A typical video portion of subsystem includes a video image compressor/decompressor (not shown) of which many are known in the art. Such compressor/decompressors convert analog video information into compressed digital information. The compressed digital information may be stored on hard disk for use at a later time. An example of such a compressor/decompressor is described in U.S. Patent No. 5,355,450.

One or more output devices may be connected to the computer system. Example output devices include a cathode ray tube (CRT) display, liquid crystal displays (LCD) and other video output devices, printers, communication devices such as a modem, storage devices such as disk or tape, and audio output. One or more input devices may be connected to the editing or playback system. Example input devices include a keyboard, keypad, track ball, mouse, pen and tablet, communication device, and data input devices such as audio and video capture devices and sensors. Source and color modification are not limited to the particular input or output devices used in combination with the computer system or to those described herein.

The computer system may be a general purpose computer system which is programmable using a computer programming language, such as "C++," JAVA or other language, such as a scripting language or even assembly language. The computer system may also be specially programmed, special purpose hardware such as an application specific integrated circuit (ASIC). In a general purpose computer system, the processor is typically a commercially available processor, such as the series x86 and Pentium processors, available from Intel, similar devices from AMD and Cyrix, the 680X0 series microprocessors available from Motorola, and the PowerPC microprocessor from IBM. Many other processors are available. Such a microprocessor executes a program called an operating system, of which WindowsNT, Windows95 or 98, UNIX, Linux, DOS, VMS, MacOS and OS8 are examples, which controls the execution of other computer programs and provides scheduling, debugging, input/output control, accounting, compilation, storage assignment, data management and memory management, and communication control and related services. The processor and operating system define a computer platform for which application programs in high-level programming languages are written.

A memory system typically includes a computer readable and writeable nonvolatile recording medium, of which a magnetic disk, a flash memory and tape are examples. The disk may be removable, known as a floppy disk, or permanent, known as a hard drive. A disk has a number of tracks in which signals are stored, typically in binary form, i.e., a form interpreted as a sequence of one and zeros. Such signals may define an application program to be executed by the microprocessor, or information stored on the disk to be processed by the application program. Typically, in operation, the processor causes data to be read from the nonvolatile recording medium into an integrated circuit memory element, which is typically a volatile, random access memory such as a dynamic random access memory (DRAM) or static memory (SRAM). The integrated circuit memory element allows for faster access to the information by the processor than does the disk. The processor generally manipulates the data within the integrated circuit memory and then copies the data to the disk after processing is completed. A variety of mechanisms are known for managing data movement between the disk and the integrated circuit memory element, and the invention is not limited thereto. It should also be understood that the invention is not limited to a particular memory system.

Such a system may be implemented in software or hardware or firmware, or a combination of the three. The various elements of the system, either individually or in combination may be implemented as a computer program product tangibly embodied in a machine-readable storage device for execution by a computer processor. Various steps of the process may be performed by a computer processor executing a program tangibly embodied on a computer-readable medium to perform functions by operating on input and generating output. Computer programming languages suitable for implementing such a system include procedural programming languages, object-oriented programming languages, and combinations of the two.

The computer system used to implement safe color modification is not limited to a particular computer platform, particular processor, or particular programming language. Additionally, the computer system may be a multi processor computer system or may include multiple computers connected over a computer network. Each step of FIGS. 4, 5, 7-13, and 16 may be separate modules of a computer program, or may be separate computer programs. Such modules may be operable on separate computers.

Having now described some embodiments, it should be apparent to those skilled in the art that the foregoing is merely illustrative and not limiting, having been presented by way of example only. Numerous modifications and other embodiments are within the scope of one of ordinary skill in the art and are contemplated as falling within the scope of the invention.

CLAIMS

1. A method of generating components for a pixel to provide safe color from components of the pixel according to a threshold defining safe color, wherein the components include a luma component, a first chroma component, and a second chroma component, the method comprising:

determining whether a combination of a first luma value associated with the luma component and a first chroma value associated with the first and second chroma components of the pixel exceeds the threshold defining safe color; and

if the combination exceeds the threshold defining safe color, modifying the luma component of the pixel so as to generate safe components.

2. The method of claim 1, further comprising:

if the combination exceeds the threshold defining safe color, modifying the first and second chroma components of the pixel so as to generate components within the threshold defining safe color.

3. The method of claim 2, further comprising:

if the combination exceeds the threshold defining safe color, generating a chroma correction scalar value,

wherein the step of modifying the first and second chroma components includes:

combining the first chroma component and the chroma correction scalar value; and

combining the second chroma component and the chroma correction scalar value.

4. The method of claim 3, wherein if the combination of the first luma value and the first chroma value is less than the threshold defining safe color, the chroma correction scalar is equal to numerical one.

5. The method of claim 1, further comprising:

if the combination exceeds the threshold defining safe color, generating a luma correction offset value,

wherein the luma component is modified by combining the luma component and the luma correction offset value.

6. The method of claim 5, wherein, the first luma value is a magnitude of a second luma value, and wherein if the second luma value is a positive value the luma correction offset value is a positive value, and if the second luma value has a negative value, the luma correction offset value is negative.

7. The method of claim 5, wherein, if the combination of the first luma value and the first chroma value is less than the threshold defining safe color, the luma correction offset value is numerical zero.

8. The method of claim 5, further comprising:

if the combination exceeds the threshold defining safe color, generating a chroma correction scalar value; and

modifying the first and second chroma components, including:

combining the first chroma component and the chroma correction scalar value; and

combining the second chroma component and the chroma correction scalar value.

9. The method of claim 1, further comprising:

receiving safe color parameters defining parameters for the components to be generated;

generating luma conversion terms from the safe color parameters; and

generating the first luma value by applying the luma conversion terms to the luma component.

10. The method of claim 9, wherein the luma conversion terms include a luma scalar value and a luma offset value.

11 The method of claim 1, further comprising:

combining the first luma value and the threshold defining safe color to produce a chroma threshold value; and

generating the first chroma value, including:

comparing a combination of the first and second chroma components to the chroma threshold value; and

if the combined chroma components is greater than the chroma threshold value, clipping the combined component value to the chroma threshold value.

12. The method of claim 11, further comprising:

generating chroma conversion terms from the safe color parameters;

generating a first chroma conversion value and a second chroma conversion value by applying the chroma conversion terms to the first and second chroma components, respectively; and

generating a combined chroma value by combining the first and second chroma conversion values,

wherein the combined chroma value is the combination of the first and second chroma components that is compared to the chroma threshold value.

13. The method of claim 1, further comprising:

receiving a luma high threshold value that defines a high threshold for the luma component;

receiving a luma low threshold value that defines a low threshold value for the luma component;

generating a checked luma value, including:

comparing the luma component and to the high luma threshold value and the low threshold value; and

if the luma component is greater than the luma high threshold value or less than the luma low threshold value, clipping the luma component to the luma high or luma low threshold value, respectively;

and

if the luma threshold value is not greater than the luma high threshold value or less than the luma low threshold value, setting the checked luma component equal to the luma component,

wherein the step of modifying the luma component modifies the checked luma component.

14. The method of claim 1, wherein the step of determining includes:

combining the first chroma value and a magnitude of the first luma value to generate the combination.

15. The method of claim 1, wherein if the combination of the first luma value and the first chroma value of the pixel exceeds the threshold defining safe color, the method further comprises:

generating an interrupt to warn a system user that an unsafe color has been detected.

16. The method of claim 1, wherein if the combination of the first luma value and the first chroma value of the pixel exceeds the threshold defining safe color, the method further comprises:

generating an event indicating that an unsafe color has been detected.

17. A method of generating corrections for modifying components of a pixel to produce a safe color, according to a threshold defining safe color, the method comprising:

transforming the color components into safe space;

determining whether a combination of transformed color components exceeds the threshold defining safe color; and

if the combination exceeds the threshold defining safe color, generating component correction values to be applied to the components and produce safe components.

18. A method of transforming a component color into safe color space, comprising:

receiving a component color of a first color space; and

transforming the component color into safe space by applying by applying conversion terms to each component of the component color,

wherein the transformation into safe color space creates symmetrical relationships between the color components, the symmetrical relationships permitting simplified operations to be performed on the components.

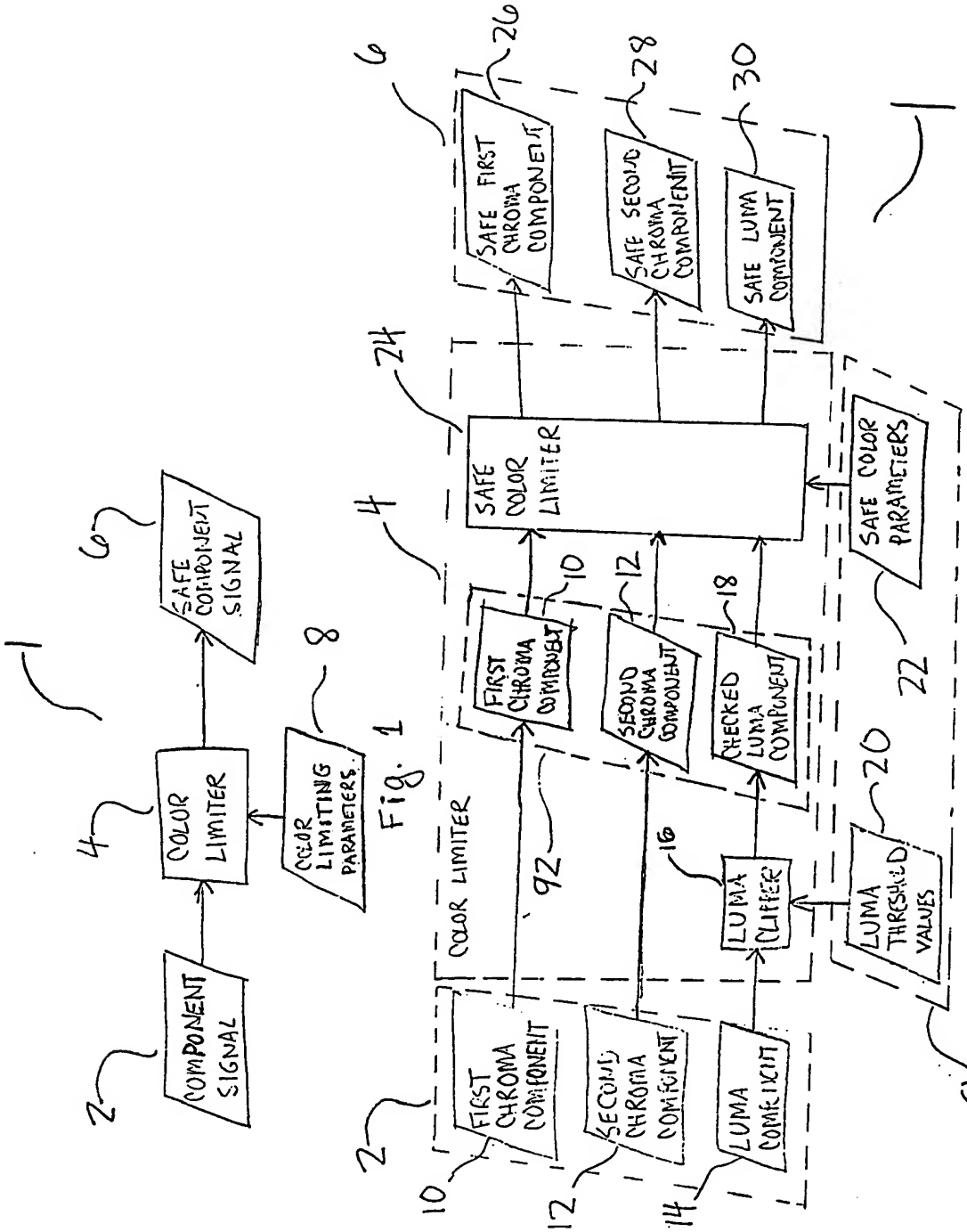

1
2
COMPONENT SIGNAL
4
COLOR LIMITER
6
SAFE COMPONENT SIGNAL
8
COLOR LIMITING PARAMETERS

Fig. 1

1
2
FIRST CHROMA COMPONENT
10
SECOND CHROMA COMPONENT
12
LUMA COMPONENT
14
COLOR LIMITER
4
FIRST CHROMA COMPONENT
10
SECOND CHROMA COMPONENT
12
CHECKED LUMA COMPONENT
18
LUMA CLIPPER
16
LUMA THRESHOLD VALUES
20
SAFE COLOR PARAMETERS
22
SAFE COLOR LIMITER
24
SAFE FIRST CHROMA COMPONENT
26
SAFE SECOND CHROMA COMPONENT
28
SAFE LUMA COMPONENT
30
6
8

Fig 2

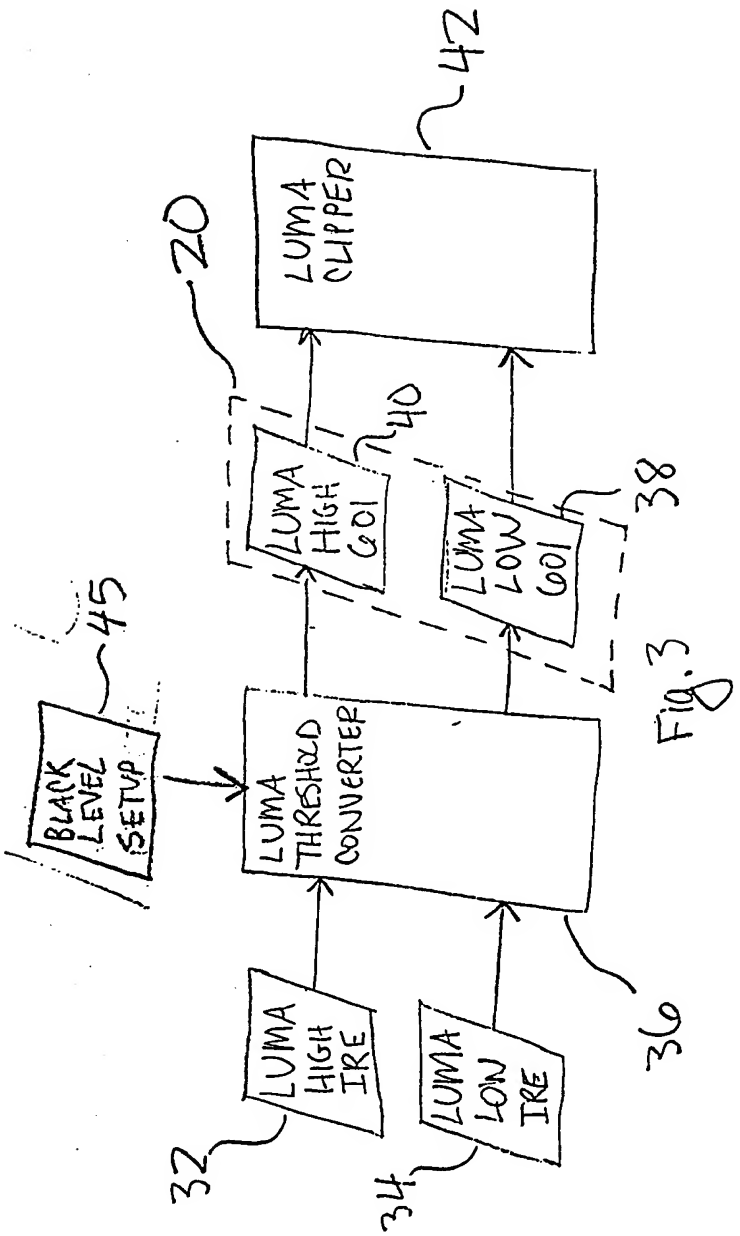

LUMA CLIPPER
42
20
LUMA HIGH GOI
40
LUMA LOW GOI
38
BLACK LEVEL SETUP
45
LUMA THRESHOLD CONVERTER
LUMA HIGH IRE
32
LUMA LOW IRE
34
36

Fig. 3

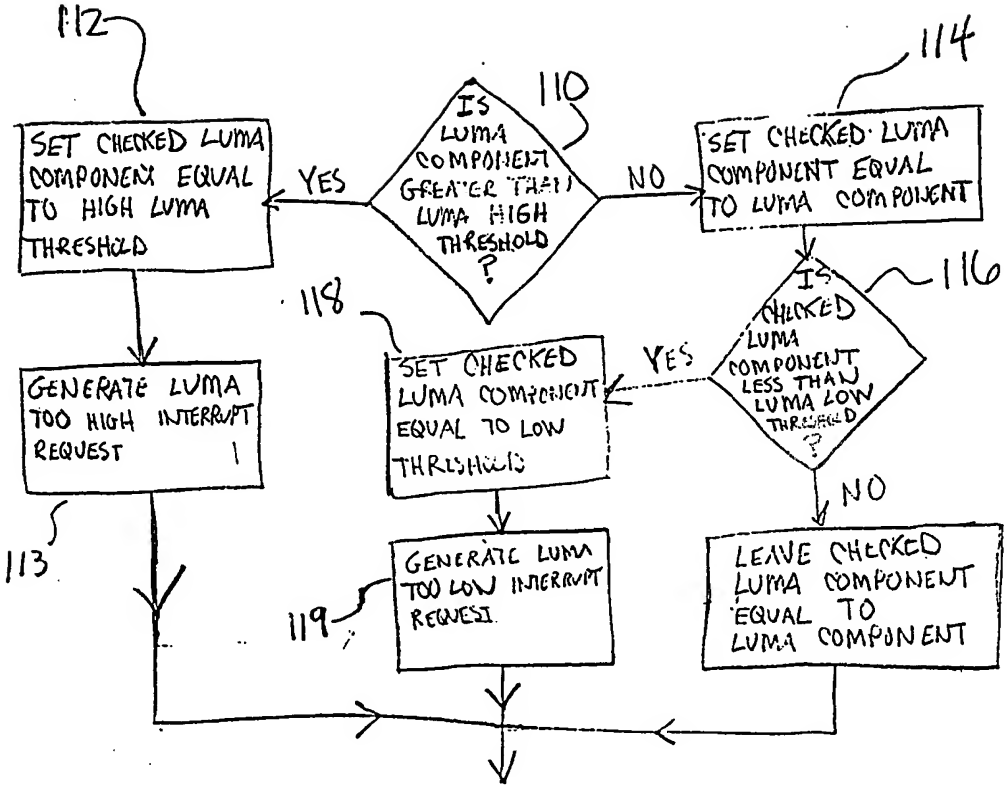

112
SET CHECKED LUMA COMPONENT EQUAL TO HIGH LUMA THRESHOLD
YES
IS LUMA COMPONENT GREATER THAN LUMA HIGH THRESHOLD ?
110
NO
114
SET CHECKED LUMA COMPONENT EQUAL TO LUMA COMPONENT
116
IS CHECKED LUMA COMPONENT LESS THAN LUMA LOW THRESHOLD ?
YES
NO
118
SET CHECKED LUMA COMPONENT EQUAL TO LOW THRESHOLD
GENERATE LUMA TOO HIGH INTERRUPT REQUEST
113
GENERATE LUMA TOO LOW INTERRUPT REQUEST
119
LEAVE CHECKED LUMA COMPONENT EQUAL TO LUMA COMPONENT

Fig. 4

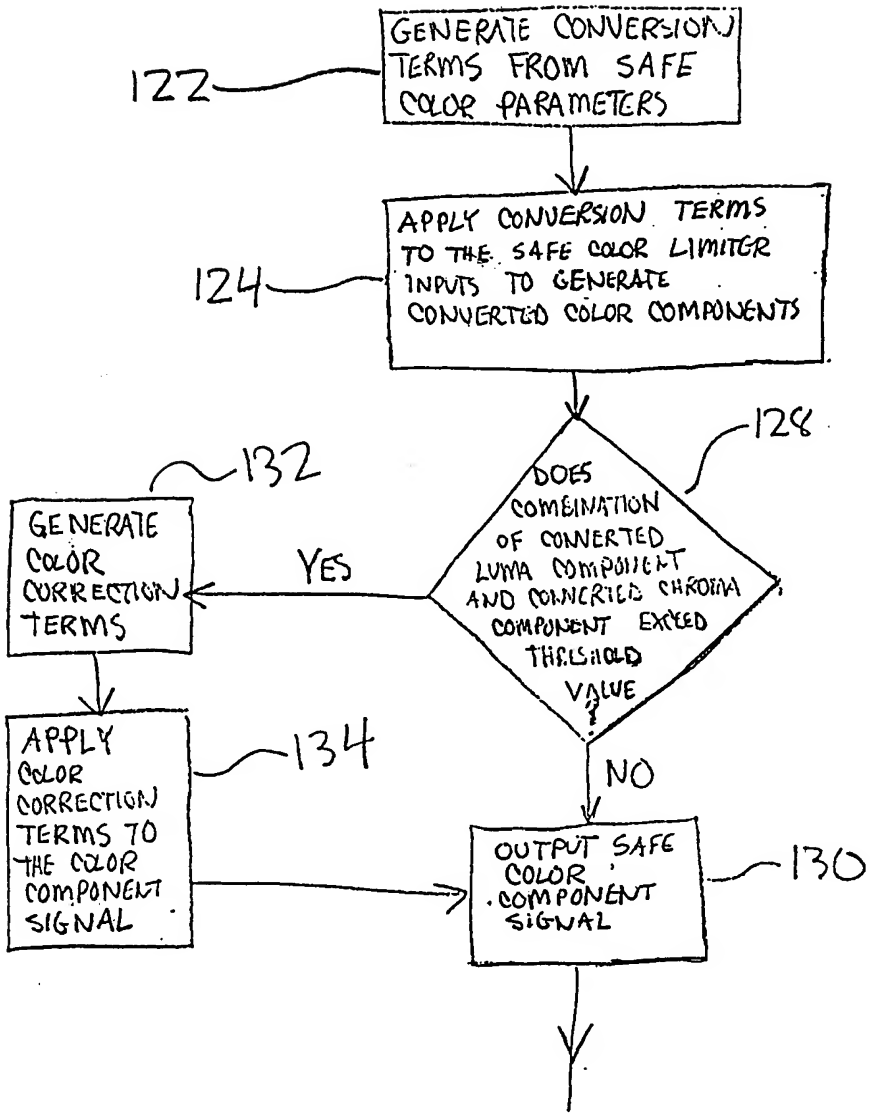

GENERATE CONVERSION TERMS FROM SAFE COLOR PARAMETERS
122
APPLY CONVERSION TERMS TO THE SAFE COLOR LIMITER INPUTS TO GENERATE CONVERTED COLOR COMPONENTS
124
DOES COMBINATION OF CONVERTED LUMA COMPONENT AND CONVERTED CHROMA COMPONENT EXCEED THRESHOLD VALUE ?
128
YES
GENERATE COLOR CORRECTION TERMS
132
APPLY COLOR CORRECTION TERMS TO THE COLOR COMPONENT SIGNAL
134
NO
OUTPUT SAFE COLOR COMPONENT SIGNAL
130

Fig. 5

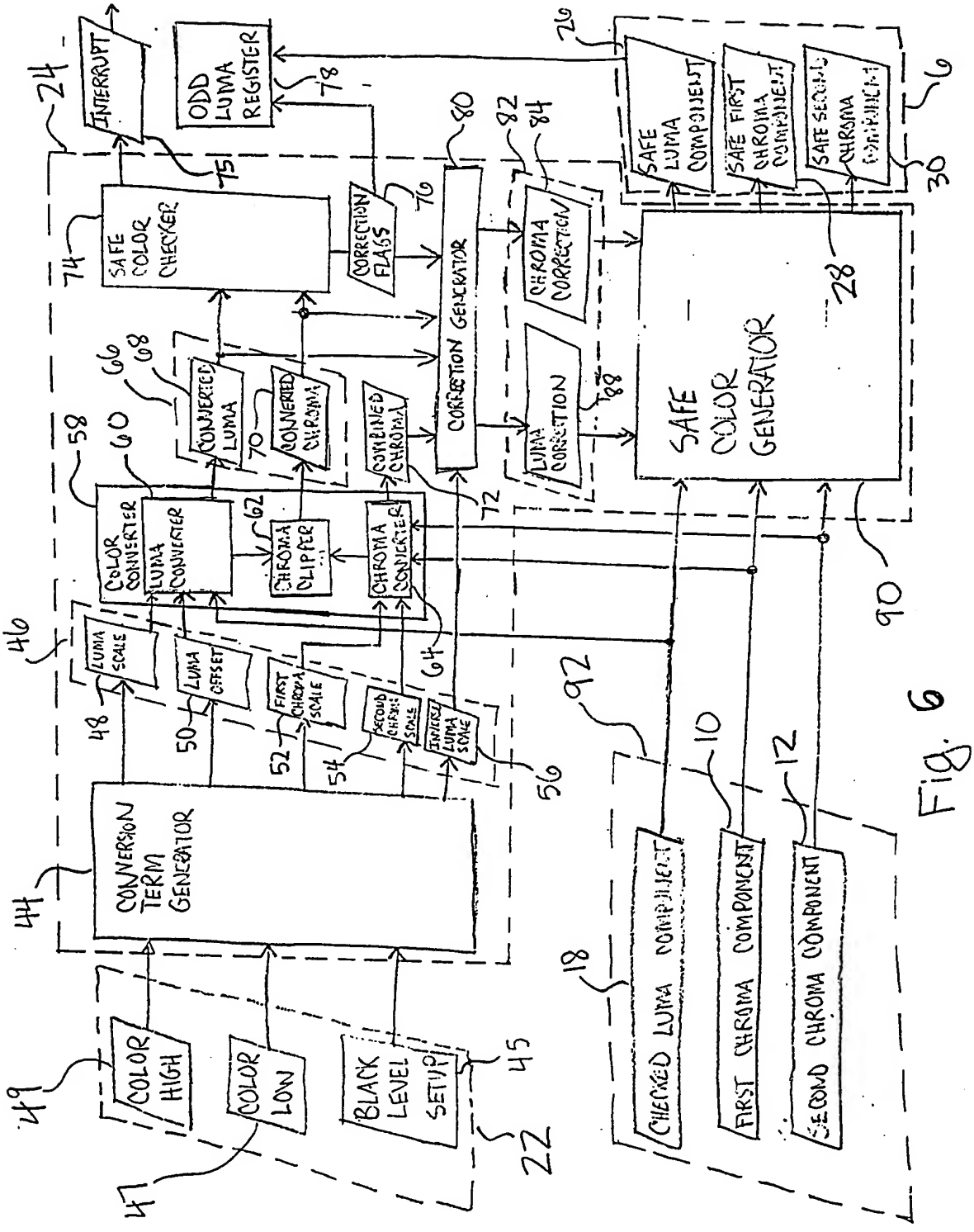

INTERRUPT
ODD LUMA REGISTER
SAFE COLOR CHECKER
CORRECTION FLAGS
CORRECTION GENERATOR
CHROMA CORRECTION
LUMA CORRECTION
SAFE LUMA COMPONENT
SAFE FIRST CHROMA COMPONENT
SAFE SECOND CHROMA COMPONENT
SAFE COLOR GENERATOR
COLOR CONVERTER
LUMA CONVERTER
CHROMA CLIPPER
CHROMA CONJUCTER
CONVERTED LUMA
CONVERTED CHROMA
COMBINED CHROMA
LUMA SCALE
LUMA OFFSET
FIRST CHROMA SCALE
SECOND CHROMA SCALE
INTERCEPT LUMA SCALE
CONVERSION TERM GENERATOR
COLOR HIGH
COLOR LOW
BLACK LEVEL SETUP
CHECKED LUMA COMPONENT
FIRST CHROMA COMPONENT
SECOND CHROMA COMPONENT
24
26
28
30
44
46
47
49
45
22
18
10
12
90
92
74
75
76
78
80
82
84
58
60
62
64
66
68
70
72
48
50
52
54
56

Fig. 9

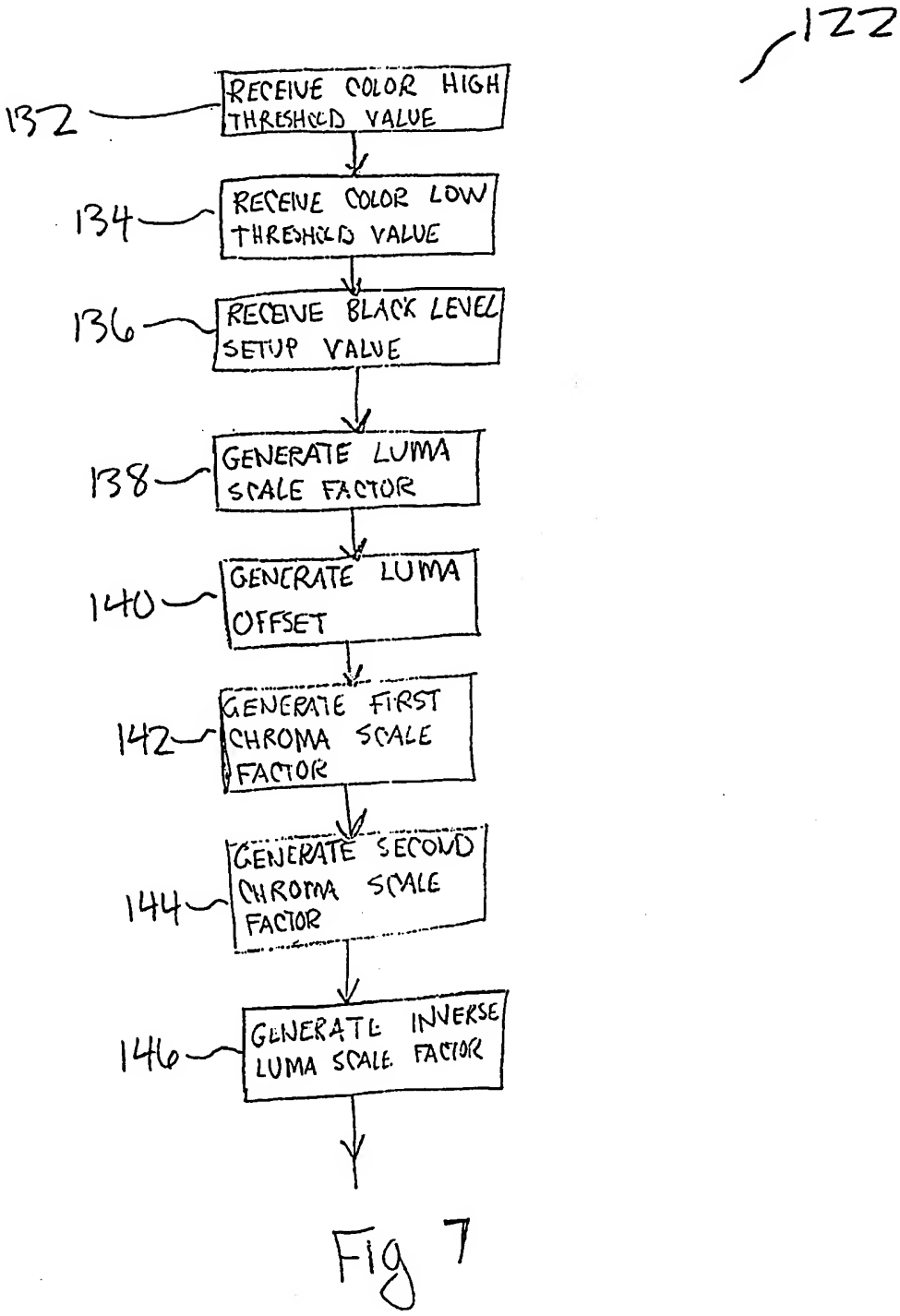

122
132
RECEIVE COLOR HIGH THRESHOLD VALUE
134
RECEIVE COLOR LOW THRESHOLD VALUE
136
RECEIVE BLACK LEVEL SETUP VALUE
138
GENERATE LUMA SCALE FACTOR
140
GENERATE LUMA OFFSET
142
GENERATE FIRST CHROMA SCALE FACTOR
144
GENERATE SECOND CHROMA SCALE FACTOR
146
GENERATE INVERSE LUMA SCALE FACTOR

Fig 7

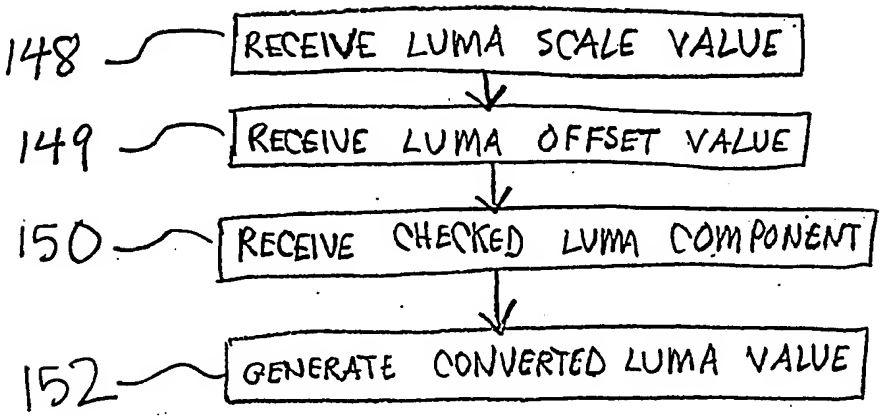

148
RECEIVE LUMA SCALE VALUE
149
RECEIVE LUMA OFFSET VALUE
150
RECEIVE CHECKED LUMA COMPONENT
152
GENERATE CONVERTED LUMA VALUE

FIG. 8

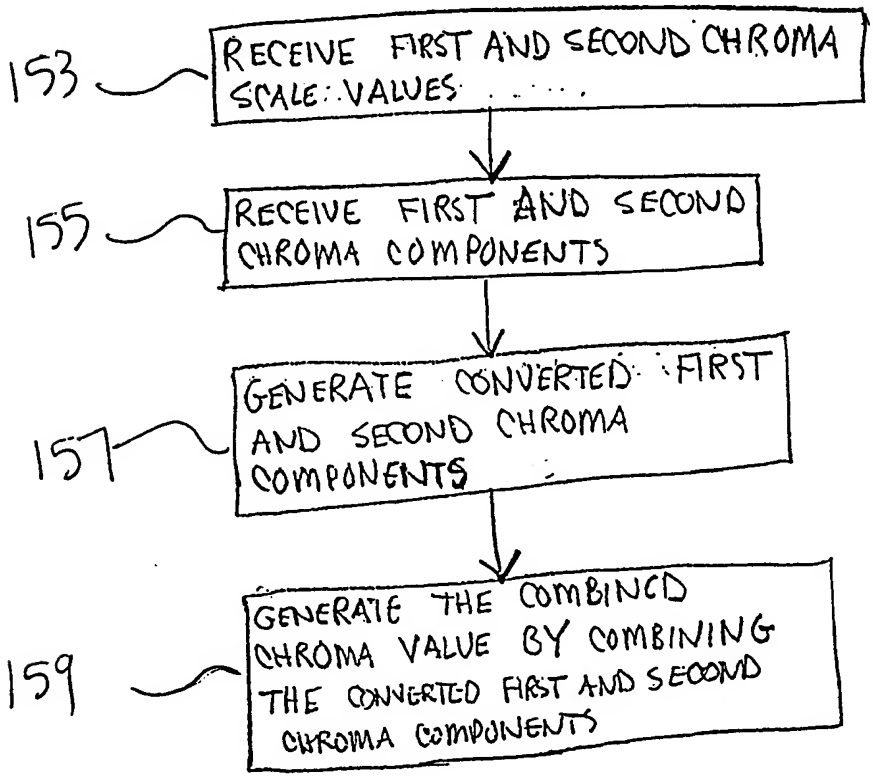

153
RECEIVE FIRST AND SECOND CHROMA SCALE VALUES
155
RECEIVE FIRST AND SECOND CHROMA COMPONENTS
157
GENERATE CONVERTED FIRST AND SECOND CHROMA COMPONENTS
159
GENERATE THE COMBINED CHROMA VALUE BY COMBINING THE CONVERTED FIRST AND SECOND CHROMA COMPONENTS

FIG. 9.

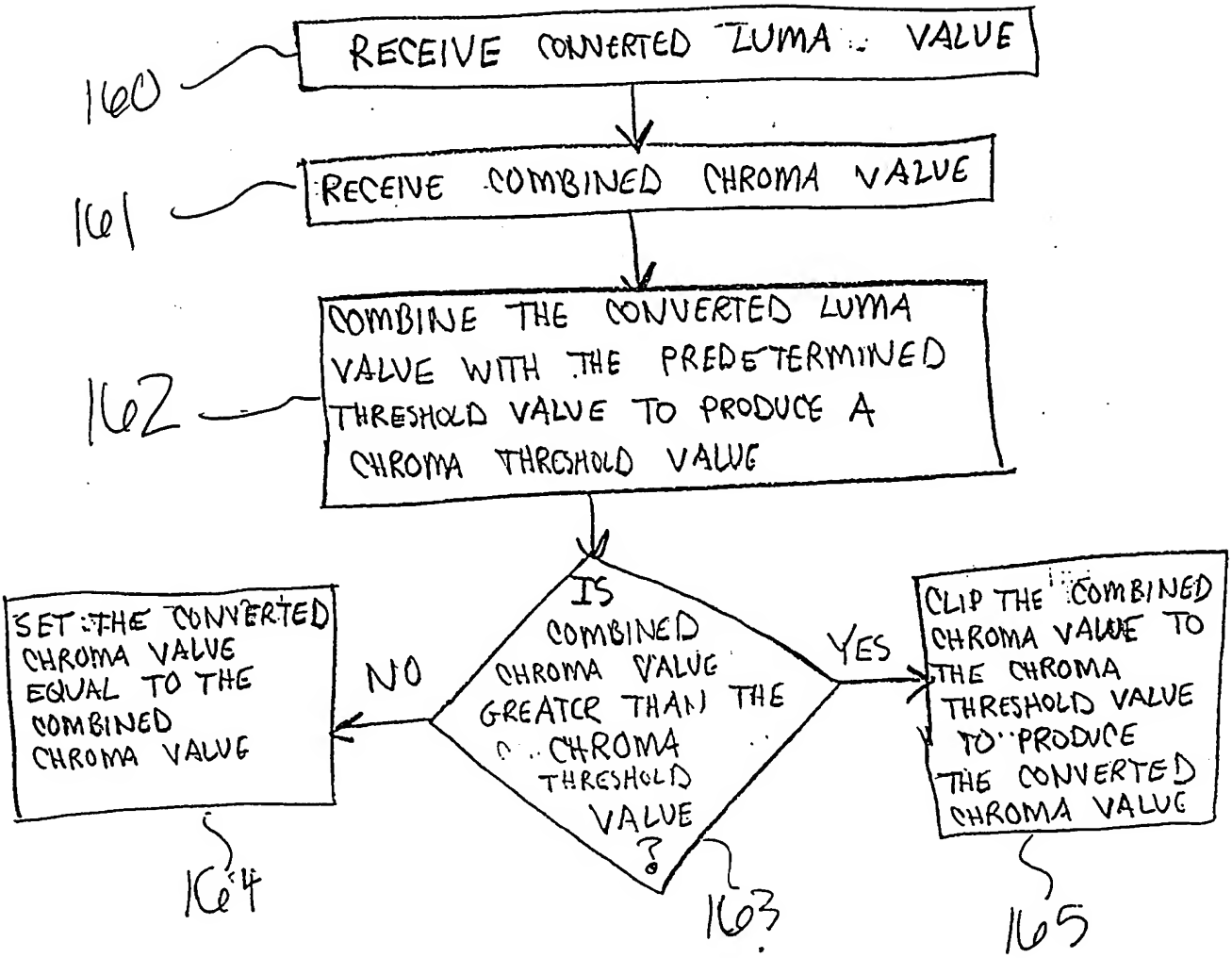

160
RECEIVE CONVERTED LUMA VALUE
161
RECEIVE COMBINED CHROMA VALUE
162
COMBINE THE CONVERTED LUMA VALUE WITH THE PREDETERMINED THRESHOLD VALUE TO PRODUCE A CHROMA THRESHOLD VALUE
IS COMBINED CHROMA VALUE GREATER THAN THE CHROMA THRESHOLD VALUE ?
163
NO
YES
SET THE CONVERTED CHROMA VALUE EQUAL TO THE COMBINED CHROMA VALUE
164
CLIP THE COMBINED CHROMA VALUE TO THE CHROMA THRESHOLD VALUE TO PRODUCE THE CONVERTED CHROMA VALUE
165

FIG. 10.

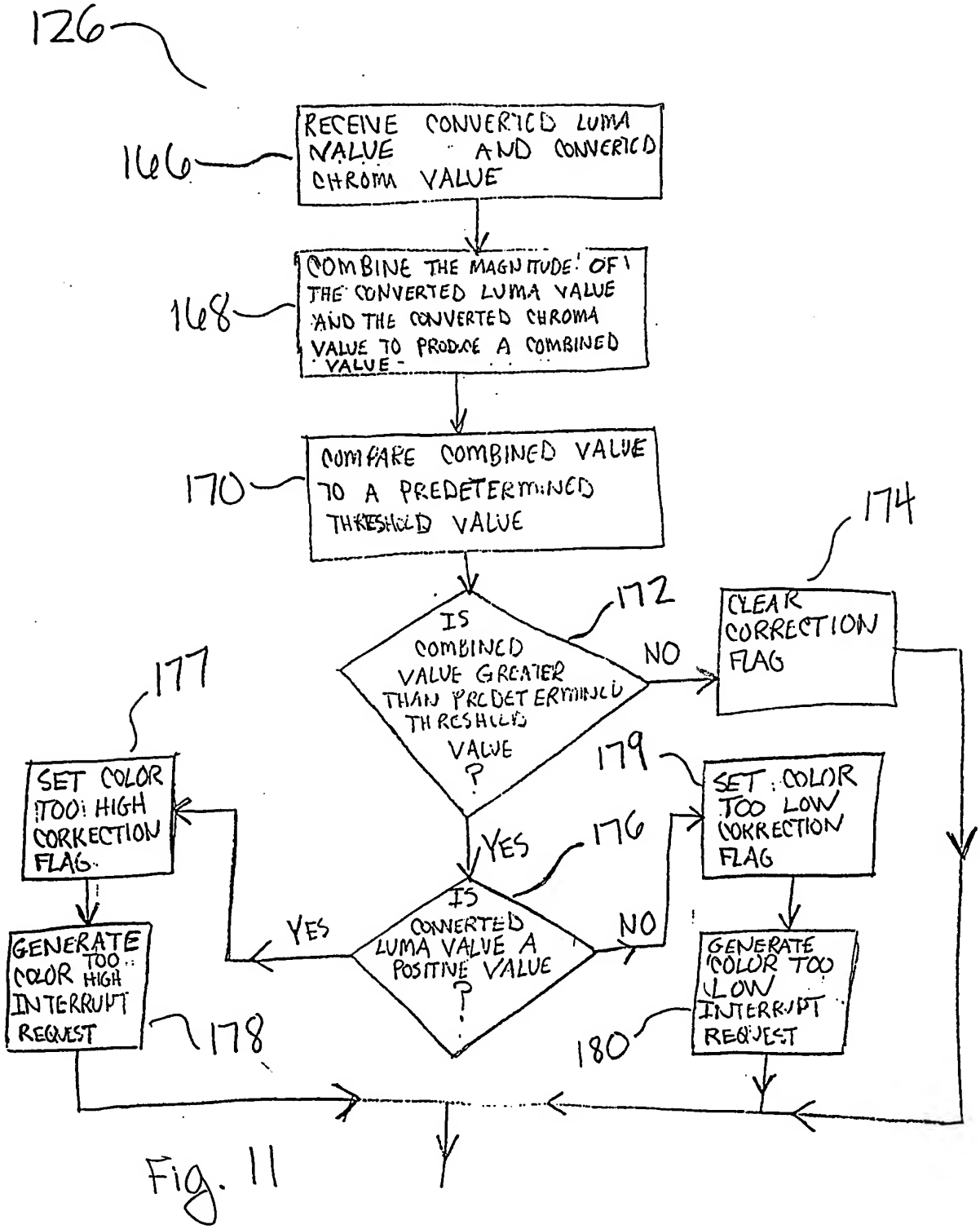

126
166
RECEIVE CONVERTED LUMA VALUE AND CONVERTED CHROMA VALUE
168
COMBINE THE MAGNITUDE OF THE CONVERTED LUMA VALUE AND THE CONVERTED CHROMA VALUE TO PRODUCE A COMBINED VALUE
170
COMPARE COMBINED VALUE TO A PREDETERMINED THRESHOLD VALUE
172
IS COMBINED VALUE GREATER THAN PREDETERMINED THRESHOLD VALUE ?
NO
174
CLEAR CORRECTION FLAG
YES
176
IS CONVERTED LUMA VALUE A POSITIVE VALUE ?
YES
177
SET COLOR TOO HIGH CORRECTION FLAG
GENERATE COLOR TOO HIGH INTERRUPT REQUEST
178
NO
179
SET COLOR TOO LOW CORRECTION FLAG
GENERATE COLOR TOO LOW INTERRUPT REQUEST
180

Fig. 11

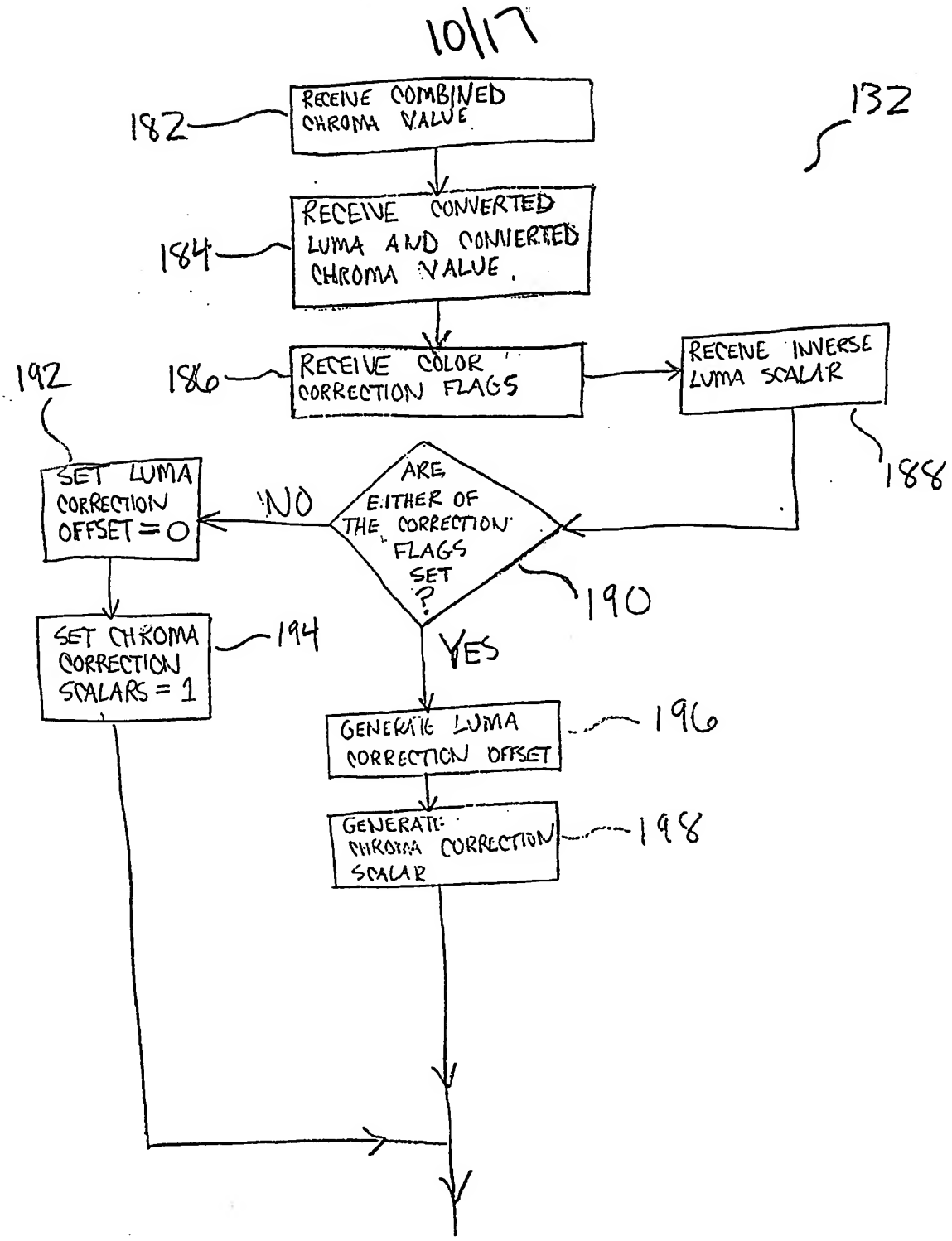

10/17
RECEIVE COMBINED CHROMA VALUE
182
132
RECEIVE CONVERTED LUMA AND CONVERTED CHROMA VALUE.
184
RECEIVE COLOR CORRECTION FLAGS
186
RECEIVE INVERSE LUMA SCALAR
188
192
SET LUMA CORRECTION OFFSET = 0
ARE EITHER OF THE CORRECTION FLAGS SET ?
NO
190
SET CHROMA CORRECTION SCALARS = 1
194
YES
GENERATE LUMA CORRECTION OFFSET
196
GENERATE CHROMA CORRECTION SCALAR
198

Fig. 12

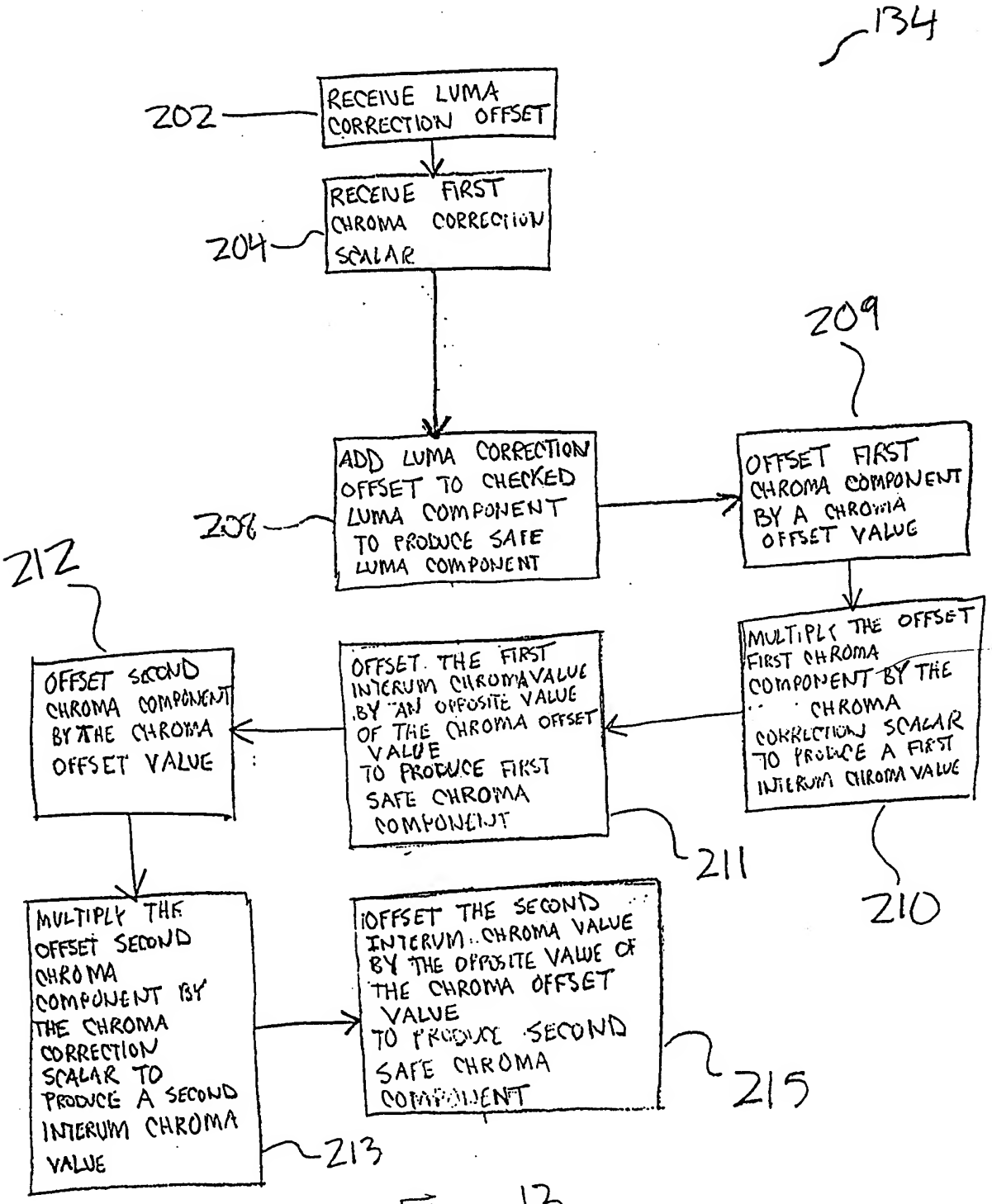

134
202
RECEIVE LUMA CORRECTION OFFSET
204
RECEIVE FIRST CHROMA CORRECTION SCALAR
206
ADD LUMA CORRECTION OFFSET TO CHECKED LUMA COMPONENT TO PRODUCE SAFE LUMA COMPONENT
209
OFFSET FIRST CHROMA COMPONENT BY A CHROMA OFFSET VALUE
MULTIPLY THE OFFSET FIRST CHROMA COMPONENT BY THE CHROMA CORRECTION SCALAR TO PRODUCE A FIRST INTERIM CHROMA VALUE
210
OFFSET THE FIRST INTERIM CHROMA VALUE BY AN OPPOSITE VALUE OF THE CHROMA OFFSET VALUE TO PRODUCE FIRST SAFE CHROMA COMPONENT
211
212
OFFSET SECOND CHROMA COMPONENT BY THE CHROMA OFFSET VALUE
MULTIPLY THE OFFSET SECOND CHROMA COMPONENT BY THE CHROMA CORRECTION SCALAR TO PRODUCE A SECOND INTERIM CHROMA VALUE
213
OFFSET THE SECOND INTERIM CHROMA VALUE BY THE OPPOSITE VALUE OF THE CHROMA OFFSET VALUE TO PRODUCE SECOND SAFE CHROMA COMPONENT
215

Fig. 13

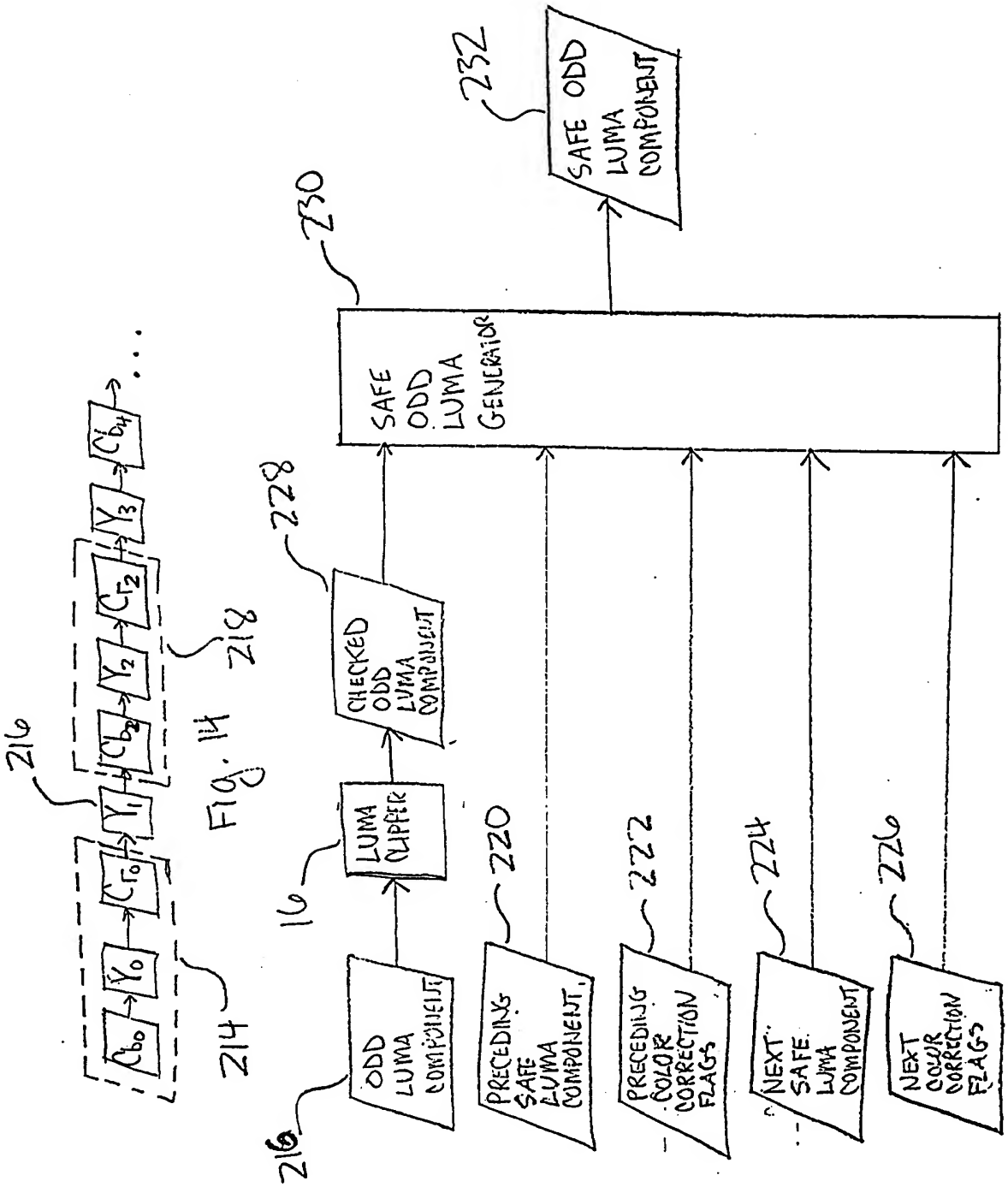

216
214
218
ODD LUMA COMPONENT
PRECEDING SAFE LUMA COMPONENT
PRECEDING COLOR CORRECTION FLAGS
NEXT SAFE LUMA COMPONENT
NEXT COLOR CORRECTION FLAGS
LUMA CLIPPER
CHECKED ODD LUMA COMPONENT
SAFE ODD LUMA GENERATOR
SAFE ODD LUMA COMPONENT
16
220
222
224
226
228
230
232

Fig. 14

Fig. 15

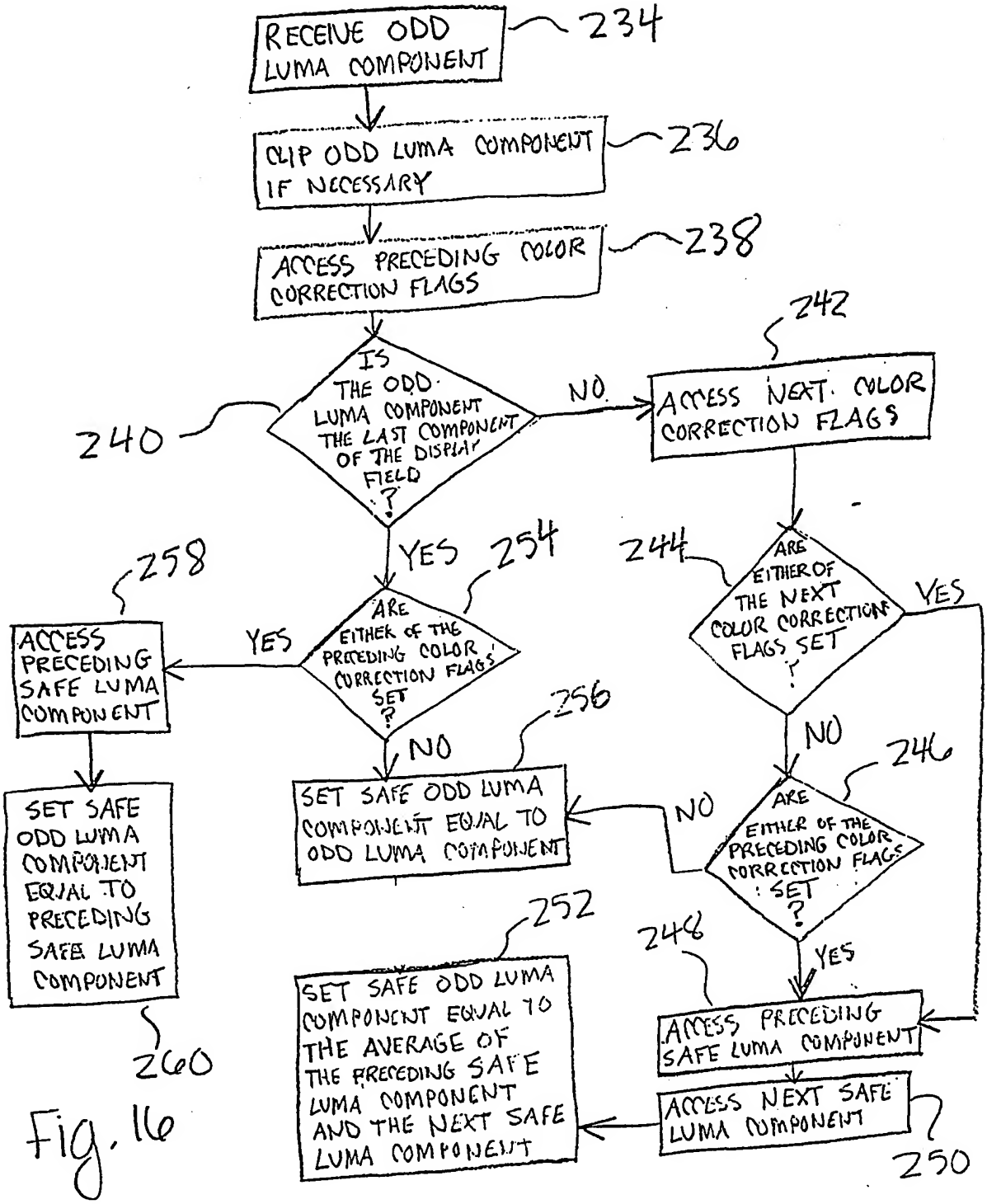

RECEIVE ODD LUMA COMPONENT
234
CLIP ODD LUMA COMPONENT IF NECESSARY
236
ACCESS PRECEDING COLOR CORRECTION FLAGS
238
IS THE ODD LUMA COMPONENT THE LAST COMPONENT OF THE DISPLAY FIELD ?
240
NO
ACCESS NEXT COLOR CORRECTION FLAGS
242
YES
ARE EITHER OF THE PRECEDING COLOR CORRECTION FLAGS SET ?
254
YES
ACCESS PRECEDING SAFE LUMA COMPONENT
258
SET SAFE ODD LUMA COMPONENT EQUAL TO PRECEDING SAFE LUMA COMPONENT
260
NO
SET SAFE ODD LUMA COMPONENT EQUAL TO ODD LUMA COMPONENT
256
ARE EITHER OF THE NEXT COLOR CORRECTION FLAGS SET ?
244
YES
NO
ARE EITHER OF THE PRECEDING COLOR CORRECTION FLAGS SET ?
246
NO
YES
ACCESS PRECEDING SAFE LUMA COMPONENT
248
ACCESS NEXT SAFE LUMA COMPONENT
250
SET SAFE ODD LUMA COMPONENT EQUAL TO THE AVERAGE OF THE PRECEDING SAFE LUMA COMPONENT AND THE NEXT SAFE LUMA COMPONENT
252

Fig. 16

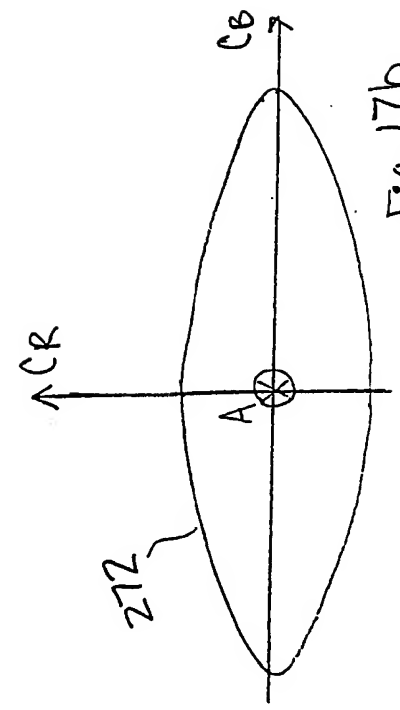


Fig. 17b

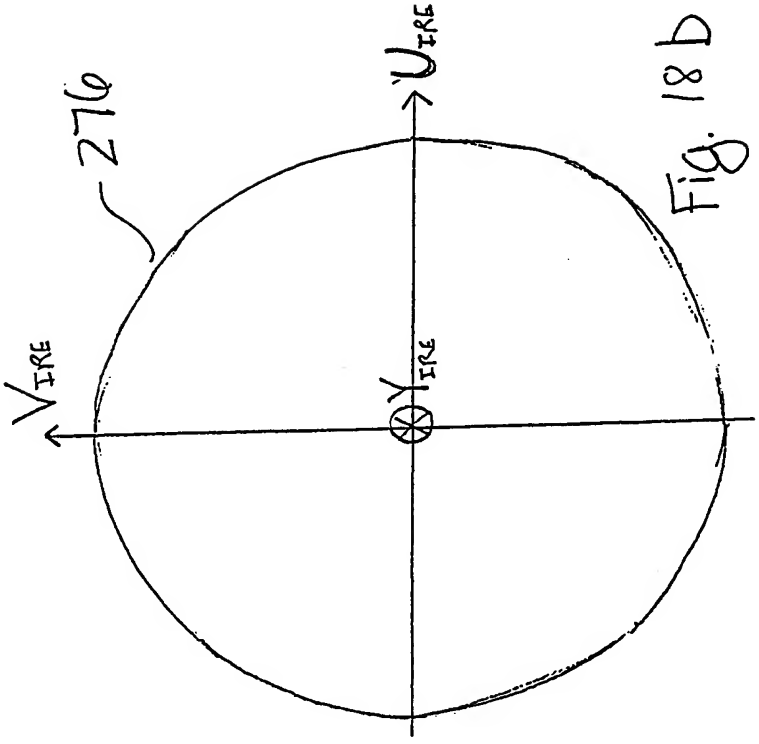


Fig. 18b

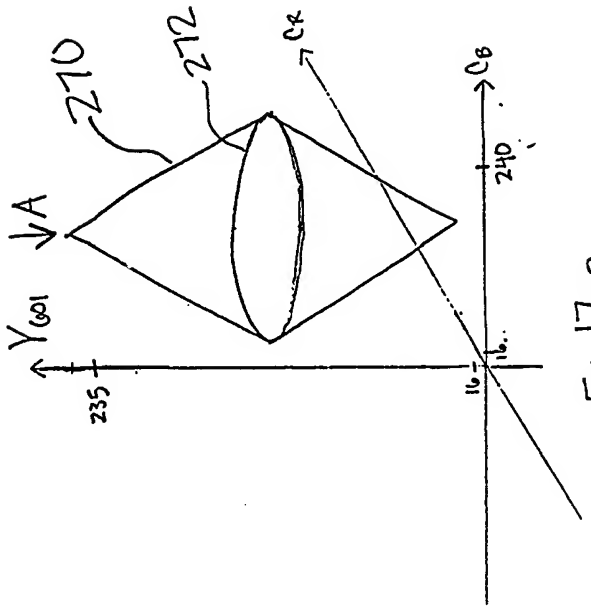


Fig 17a

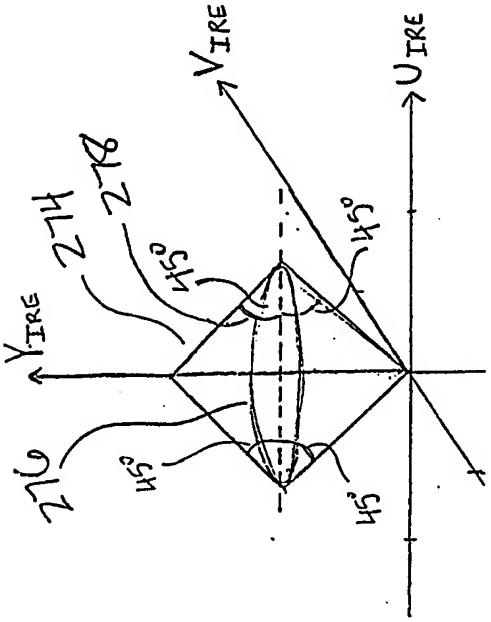
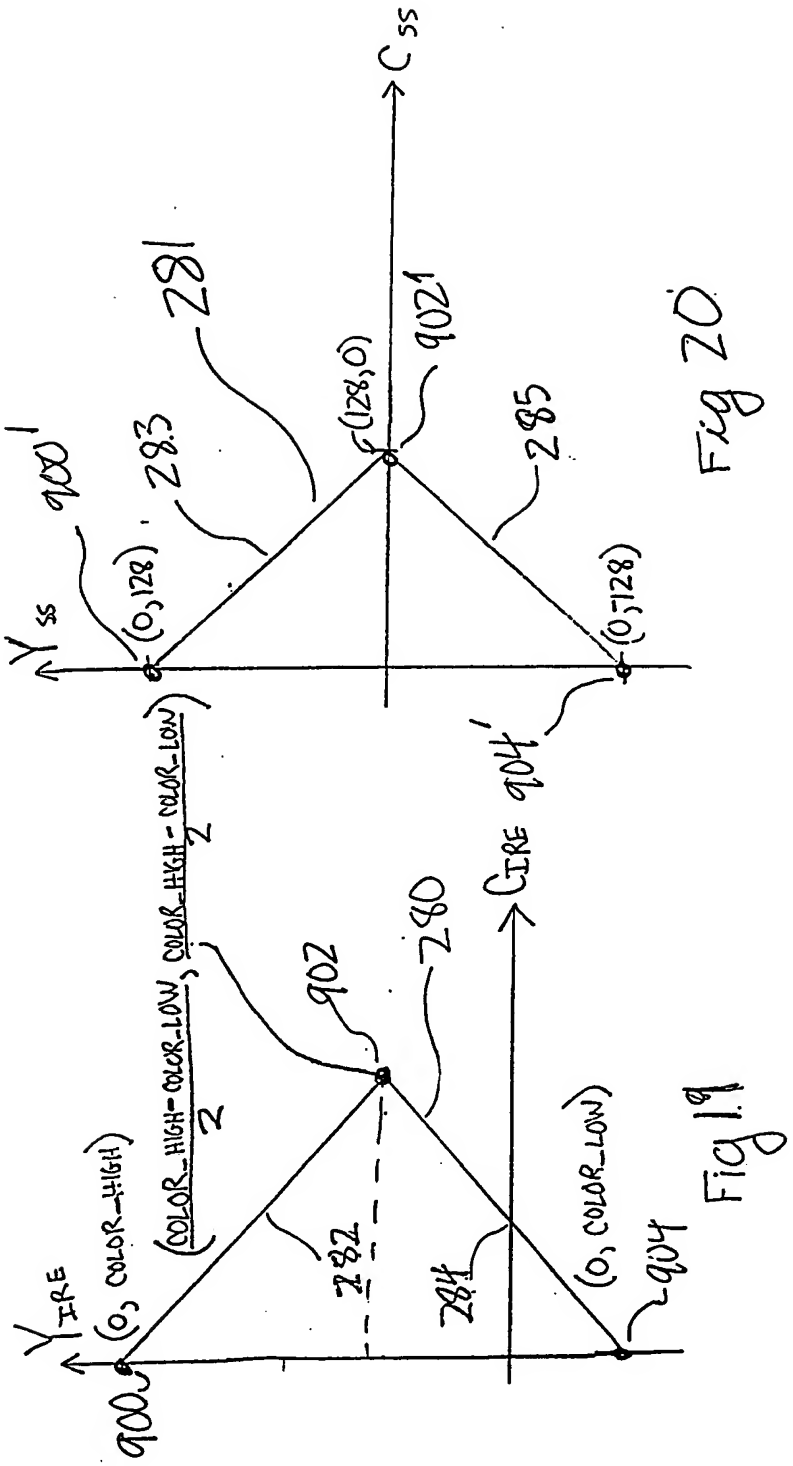


Fig 18a


Y SS
C SS
(0,128)
(128,0)
(0,-128)
900'
902'
904'
281
283
285
Fig 20
Y IRE
C IRE
(0, COLOR_HIGH)
((COLOR_HIGH-COLOR_LOW)/2, (COLOR_HIGH-COLOR_LOW)/2)
(0, COLOR_LOW)
900
902
904
280
282
284
Fig 19

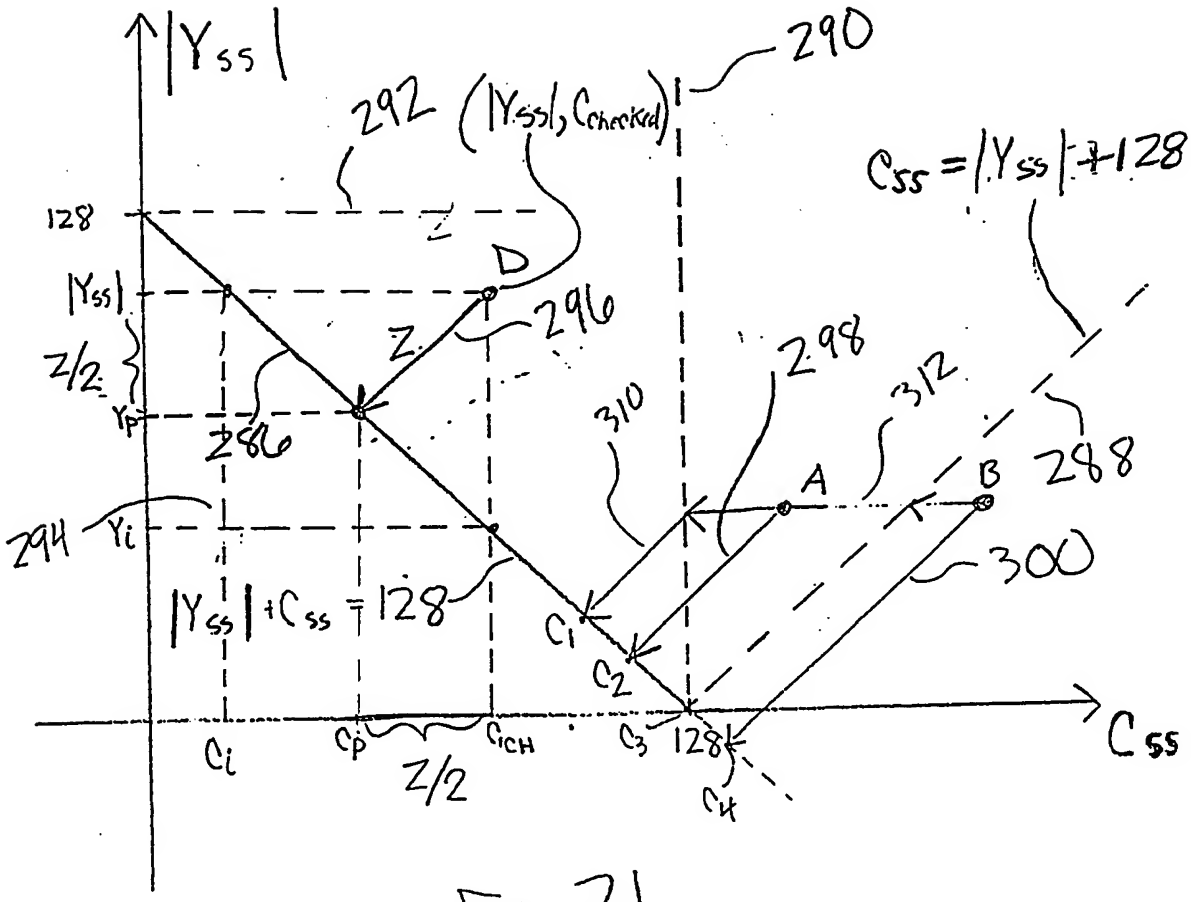

|Yss|
290
292 (|Yss|, Cchecked)
Css = |Yss| + 128
128
|Yss|
D
296
Z
Z/2
Yp
286
298
312
310
288
A
B
294
Yi
300
|Yss| + Css = 128
C1
C2
Css
Ci
Cp
Z/2
CICH
C3
128
C4

Fig 21

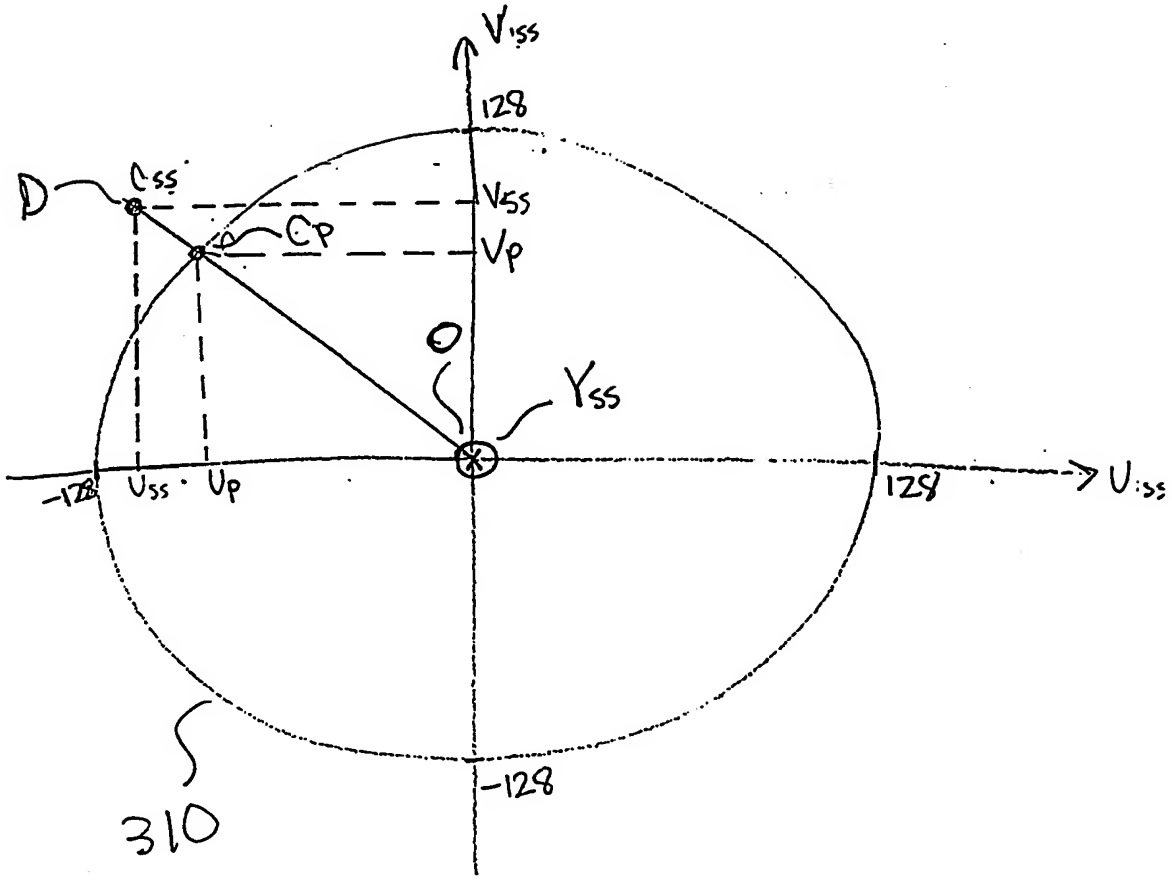


Fig. 22